

BỘ GIÁO DỤC VÀ ĐÀO TẠO  
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHÓA D20



**Báo Cáo Nhập Môn Trí Tuệ Nhân Tạo**

**Đề tài**

**Nhận diện biển số xe bằng phương pháp học sâu**

Lớp : CNPM2 - 08

Giảng viên : Trần Đình Quế

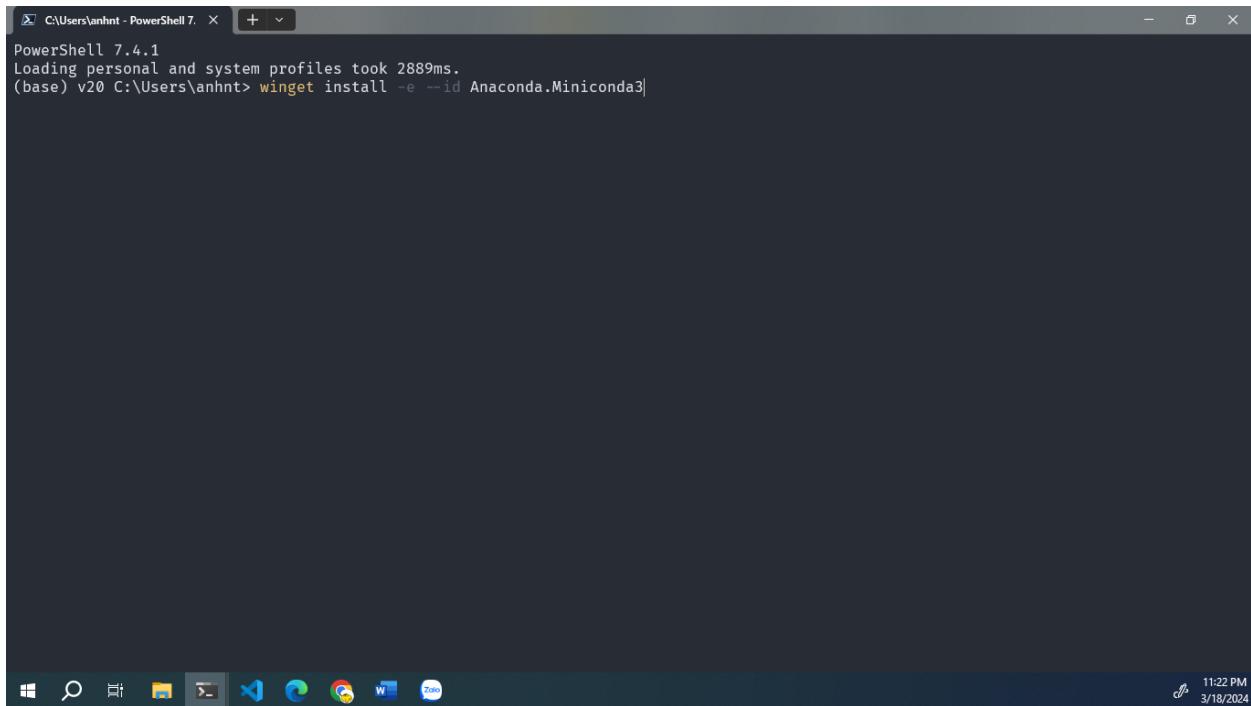
Sinh viên : Nguyễn Tiến Anh

*Hà Nội, 2023*

## I. Tạo 10 services

Tải miniconda để quản lý môi trường và các package cài về của python

```
winget install -e --id Anaconda.Miniconda3
```

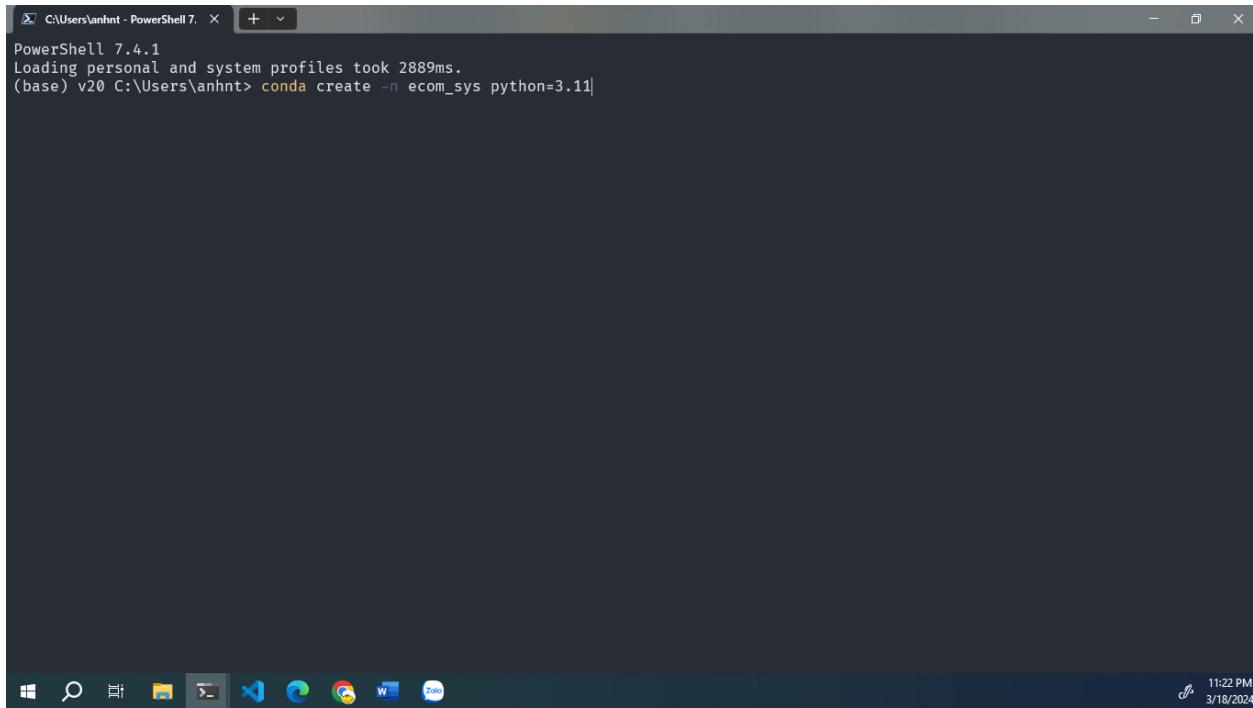


```
C:\Users\anhnt - PowerShell 7. × + ▾
PowerShell 7.4.1
Loading personal and system profiles took 2889ms.
(base) v20 C:\Users\anhnt> winget install -e --id Anaconda.Miniconda3|
```

Cài đặt miniconda theo hướng dẫn. Sau khi cài đặt xong, thêm path của miniconda vào environment variable của Windows, sử dụng dòng lệnh

```
conda create -n ecom_sys python=3.11
```

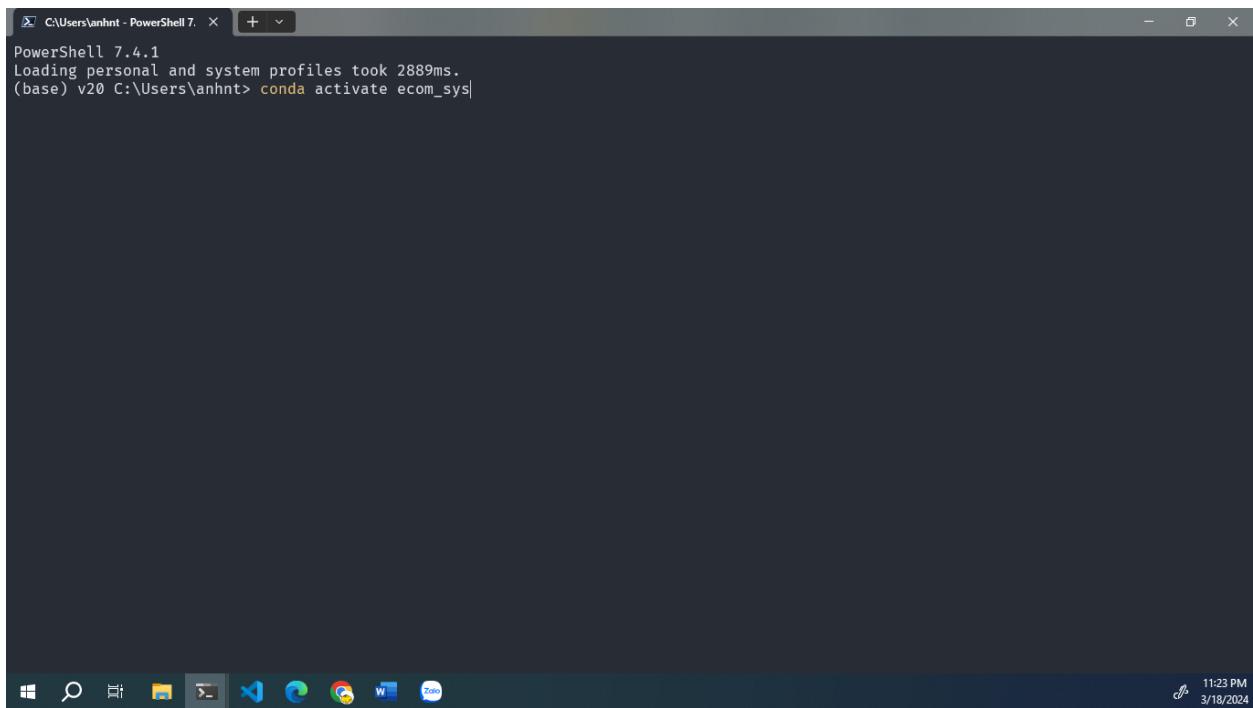
để tạo một môi trường quản lý phiên bản python (ở đây là python 3.11).



```
PowerShell 7.4.1
Loading personal and system profiles took 2889ms.
(base) v20 C:\Users\anhnt> conda create -n ecom_sys python=3.11|
```

Chuyển môi trường sang ecom\_sys bằng dòng lệnh

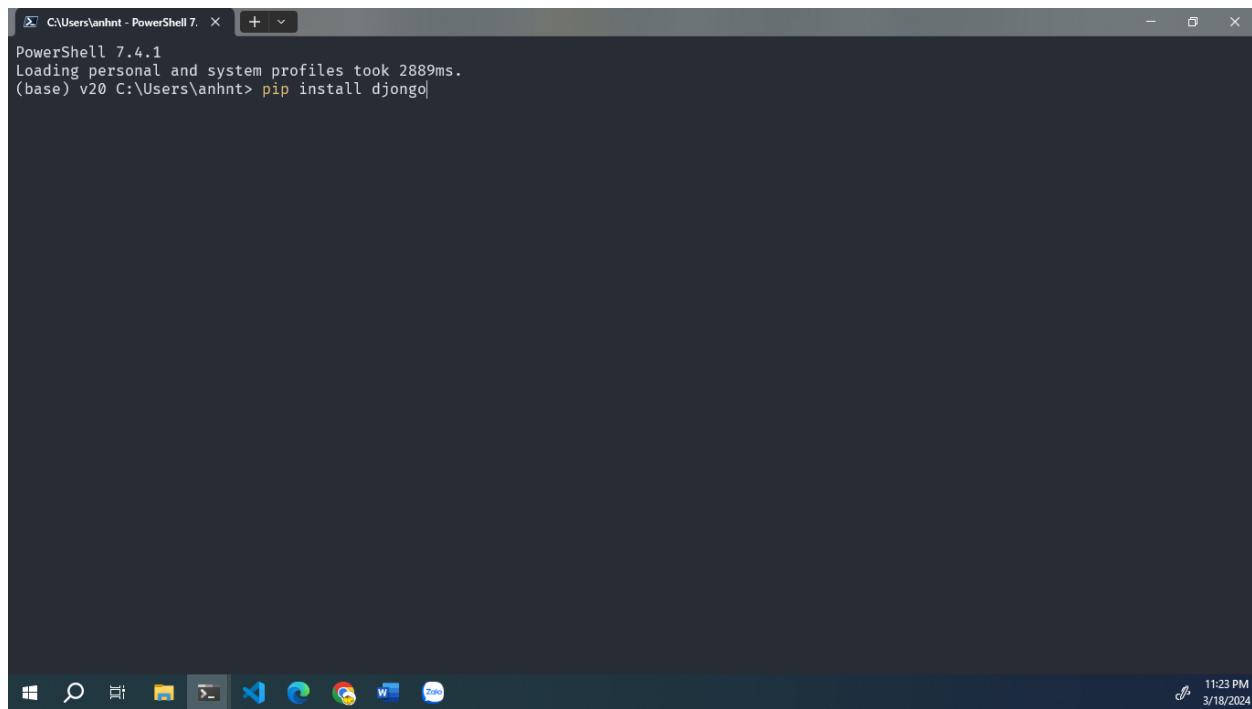
conda activate ecom\_sys



```
PowerShell 7.4.1
Loading personal and system profiles took 2889ms.
(base) v20 C:\Users\anhnt> conda activate ecom_sys|
```

Cài đặt django bằng lệnh pip install sau đây

```
pip install djongo
```



A screenshot of a Windows PowerShell window titled 'C:\Users\anhnt - PowerShell 7'. The window shows the command 'pip install djongo' being typed. The PowerShell interface includes a dark theme, a taskbar at the bottom with various icons, and a system tray on the right showing the date and time (3/18/2024, 11:23 PM).

Django được cài đặt bởi djongo sẽ phù hợp nhất cho việc phát triển ứng dụng Django có database là MongoDB.

Init project mới bằng lệnh

```
django-admin startproject ecom_sys
cd ecom_sys
```

```
C:\Users\anhnt - PowerShell 7. x +   
PowerShell 7.4.1  
Loading personal and system profiles took 2889ms.  
(base) v20 C:\Users\anhnt> django-admin startproject ecom_sys  
-> cd ecom_sys|
```

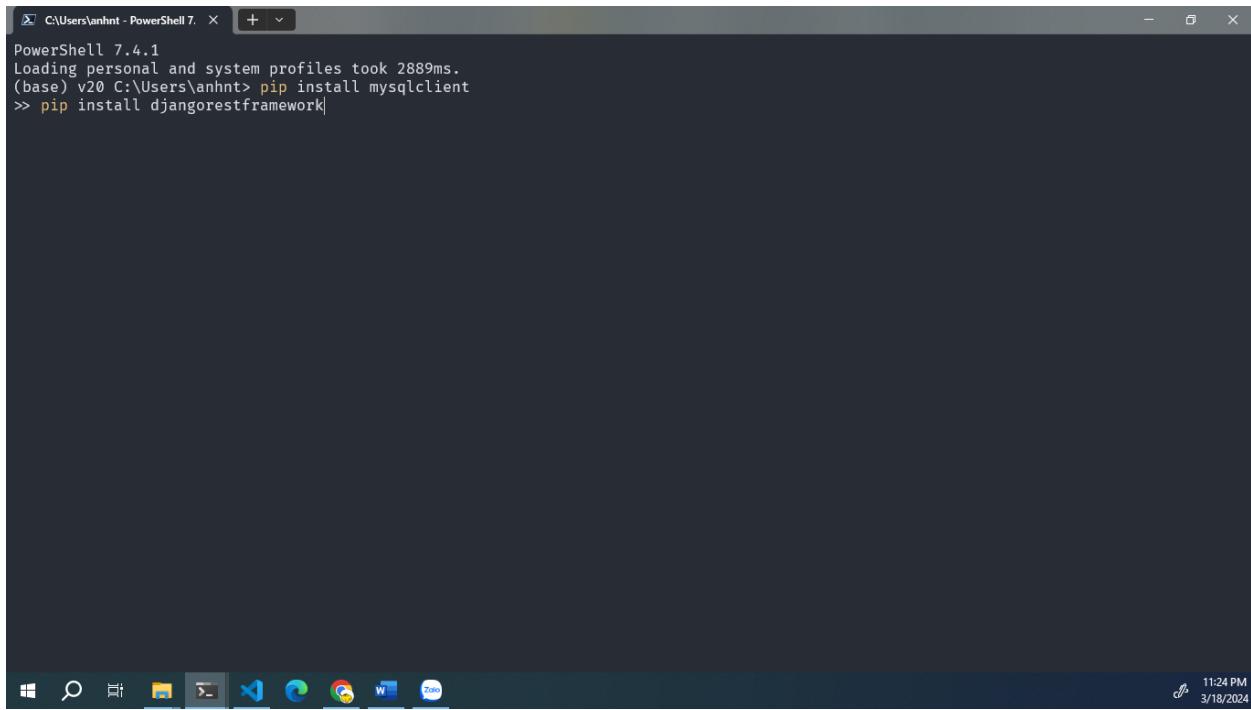
## Chạy project lần đầu bằng lệnh

```
python manage.py runserver
```

```
C:\Users\anht - PowerShell 7. x +   
PowerShell 7.4.1  
Loading personal and system profiles took 2889ms.  
(base) v20 C:\Users\anht> python manage.py runserver|
```

- Cài đặt các package sau đây:
    - mysqlclient: pip install mysqlclient

- Django rest framework: pip install djangorestframework



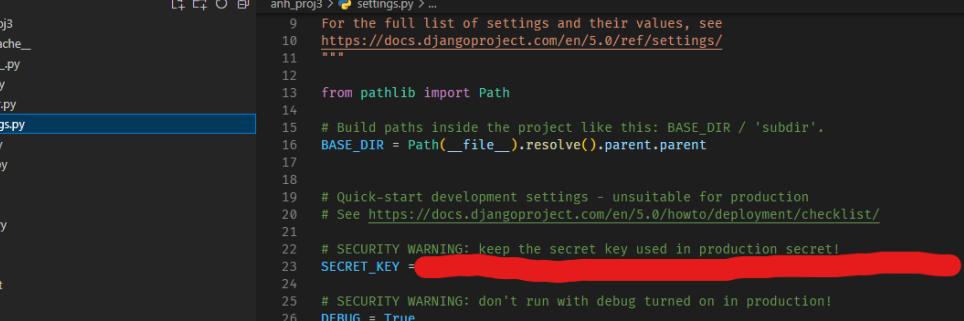
A screenshot of a Windows PowerShell window titled 'C:\Users\anhnt - PowerShell 7'. The window shows the command 'pip install djangorestframework' being typed. The PowerShell interface includes a dark theme, a taskbar at the bottom with various icons, and a system tray on the right showing the date and time (3/18/2024, 11:24 PM).

- Sau khi cài đặt các package trên, thực hiện tạo các service:
  - User: django-admin startapp user
  - Product: django-admin startapp product
  - Book: Django-admin startapp book
  - Cloth: Django-admin startapp cloth
  - Mobile: Django-admin startapp mobile
  - Category: Django-admin startapp category
  - Cart: Django-admin startapp cart
  - Search: Django-admin startapp search
  - Payment: Django-admin startapp payment
  - Shipment: Django-admin startapp shipment

```
C:\Users\anhnt - PowerShell 7. x + - □ ×  
PowerShell 7.4.1  
Loading personal and system profiles took 2889ms.  
(base) v20 C:\Users\anhnt> django-admin startapp user  
django-admin: The term 'django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> django-admin startapp product  
django-admin: The term 'django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp book  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp cloth  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp mobile  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp category  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp cart  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp search  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp payment  
Django-admin: The term 'Django-admin' is not recognized as a name of a cmdlet, function, script file, or executable program.  
Check the spelling of the name, or if a path was included, verify that the path is correct and try again.  
(base) v20 C:\Users\anhnt> Django-admin startapp shipment|
```

## II. Cấu hình RESTful cho các API

- Vào settings.py để tiện cho việc cấu hình:
    - Sửa SECRET KEY cho việc mã hóa dữ liệu

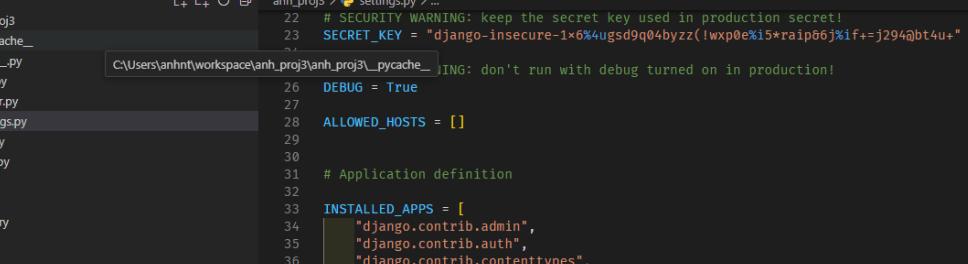


The screenshot shows a code editor with a Django project structure. The left sidebar displays the project tree under 'ANH\_PROJ3', including 'book', 'cart', 'category', 'cloth', 'mobile', 'product', 'search', 'user', and several static files like 'baitap\_3\_REST API\_chap 11.pdf', 'manage.py', and 'requirements.txt'. The right pane shows the 'settings.py' file for the 'product' app. The code includes settings for paths, security, and application definition.

```
File Edit Selection View Go Run ... ← → anh_proj3
FOLDERS
ANH_PROJ3
anh_proj3
__pycache__
    __init__.py
    asgi.py
    router.py
settings.py
    book
    cart
    category
    cloth
    mobile
    product
    search
    user
    baitap_3_REST API_chap 11.pdf
    manage.py
    requirements.txt
OUTLINE
TIMELINE
EXPLORER: NPM SCRIPTS
EXPLORER: VS CODE PETS
Ln 48, Col 14  Spaces: 4  UTF-8  CRLF  Python  3.12.1 (anh_proj):conda  Go Live  II Ninja  Prettier
11:21 PM 3/18/2024
```

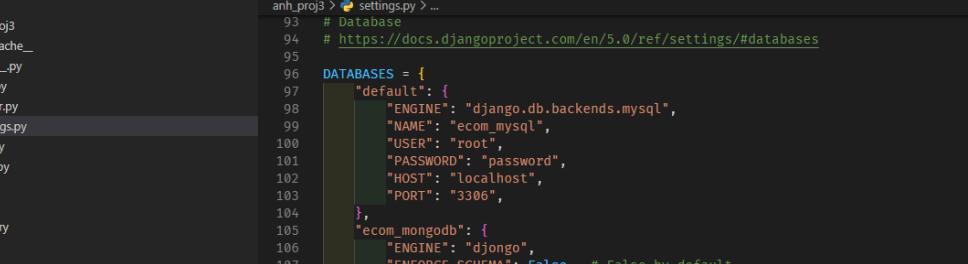
```
models.py product settings.py router.py models.py book serializers.py urls.py
anh_proj3 > settings.py > ...
9  For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/5.0/ref/settings/
11 """
12
13 from pathlib import Path
14
15 # Build paths inside the project like this: BASE_DIR / 'subdir'.
16 BASE_DIR = Path(__file__).resolve().parent.parent
17
18
19 # Quick-start development settings - unsuitable for production
20 # See https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/
21
22 # SECURITY WARNING: keep the secret key used in production secret!
23 SECRET_KEY = [REDACTED]
24
25 # SECURITY WARNING: don't run with debug turned on in production!
26 DEBUG = True
27
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     "django.contrib.admin",
35     "django.contrib.auth",
36     "django.contrib.contenttypes",
37     "django.contrib.sessions",
38     "django.contrib.messages",
39     "django.contrib.staticfiles",
40     "rest_framework",
```

- Thêm rest framework vào INSTALLED APPS



```
File Edit Selection View Go Run ... ← → anh_proj3 FOLDERS ... models.py product settings.py router.py models.py book serializers.py urls.py ... ANH_PROJS anh_proj3 anh_proj3_pycache_ ... anh_proj3 > anh_proj3_pycache_ ... C:\Users\anhnt\workspace\anh_proj3\anh_proj3_pycache_ ... JING: don't run with debug turned on in production! 22 # SECURITY WARNING: keep the secret key used in production secret! 23 SECRET_KEY = "django-insecure-1x6%ugsd9q04byzz(!wxp0e%15*raip66j%if+=j294@bt4u+" 24 25 DEBUG = True 26 27 28 ALLOWED_HOSTS = [] 29 30 31 # Application definition 32 33 INSTALLED_APPS = [ 34     "django.contrib.admin", 35     "django.contrib.auth", 36     "django.contrib.contenttypes", 37     "django.contrib.sessions", 38     "django.contrib.messages", 39     "django.contrib.staticfiles", 40     "rest_framework", 41     # "drf_spectacular", 42     "category", 43     "book", 44     "mobile", 45     "cloth", 46     "product", 47     # "user", 48     # "cart", 49 ] 50 51 MIDDLEWARE = [ 52     "django.middleware.security.SecurityMiddleware", 53     "django.contrib.sessions.middleware.SessionMiddleware", 54     "django.middleware.common.CommonMiddleware", 55     "django.middleware.csrf.CsrfViewMiddleware", 56     "django.middleware.clickjacking.XFrameOptionsMiddleware", 57 ]
```

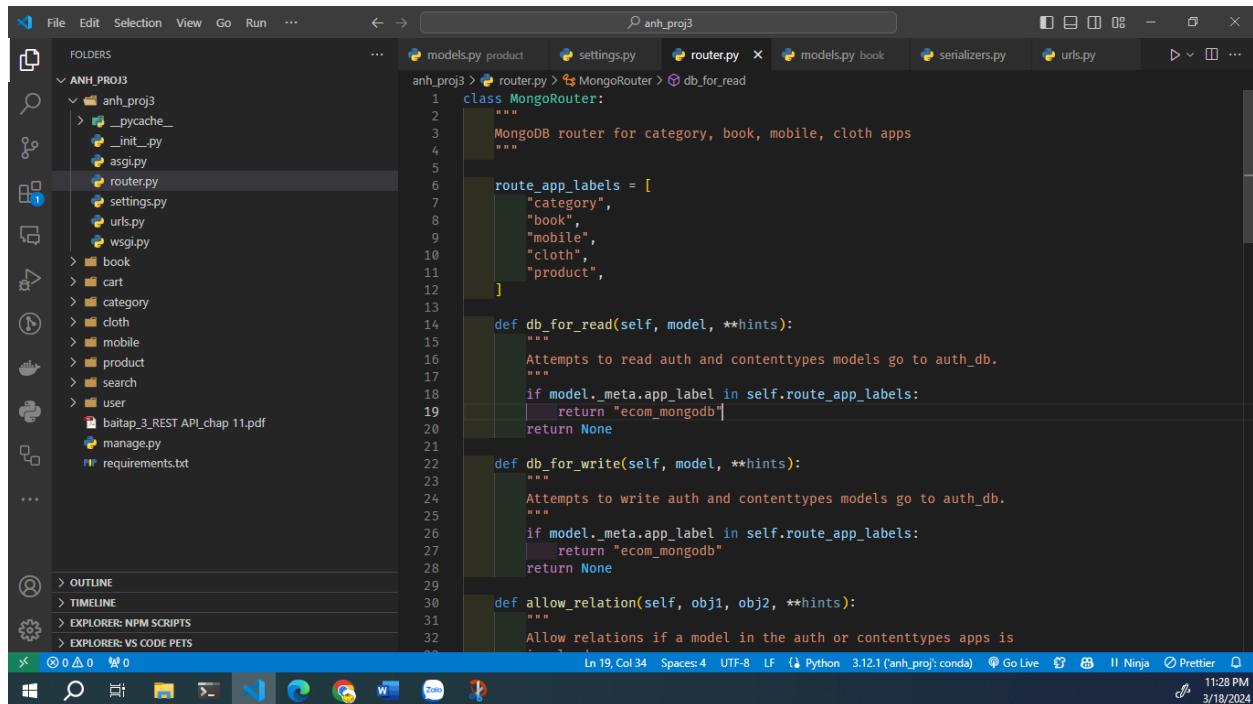
- Cấu hình database cho tiện cho việc phát triển app dựa trên MongoDB và MySQL.



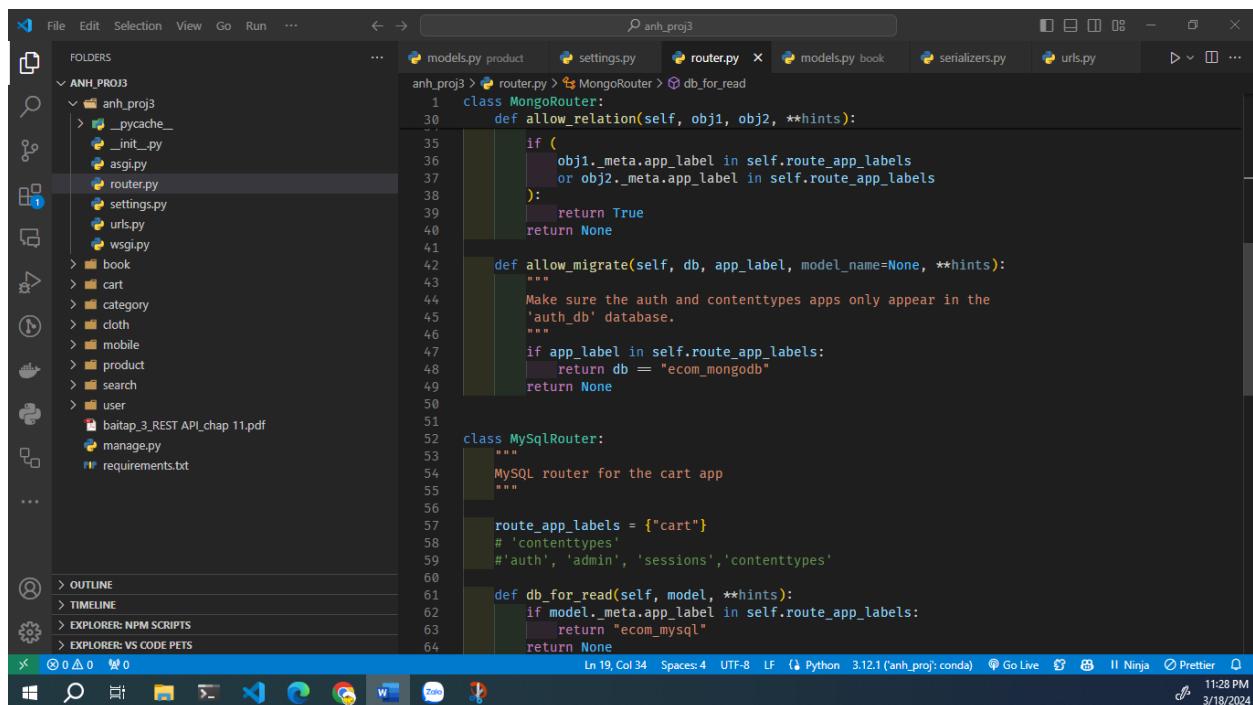
The screenshot shows a Python IDE interface with the following details:

- File Explorer (Left):** Shows the project structure under "ANH\_PROJ".
  - Subfolders: anh\_proj3, \_pycache\_, book, cart, category, cloth, mobile, product, search, user.
  - Files: models.py (product), settings.py (active), router.py, models.py (book), serializers.py, urls.py.
  - Others: baitap\_3\_REST API\_chap 11.pdf, manage.py, requirements.txt.
- Code Editor (Center):** Displays the content of `settings.py`. The code defines the `DATABASES` and `DATABASE_ROUTERS` settings for a Django project using MySQL and MongoDB.
- Bottom Status Bar:** Shows the current file is `settings.py`, line 119, column 29. Other status indicators include Python 3.12.1, Go Live, II Ninja, and Prettier.

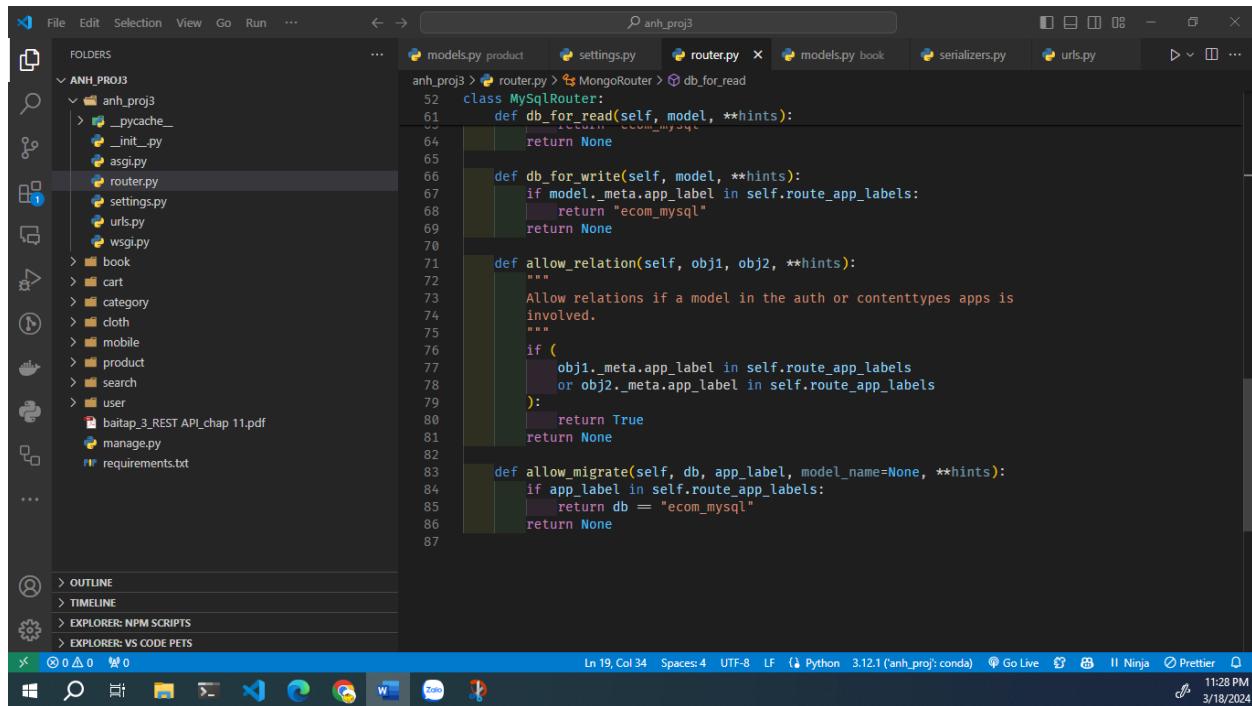
- Tạo file routers.py để cấu hình điều hướng các app được tạo sao cho từng app dùng đúng loại database mà người dùng muốn.



```
1  class MongoRouter:
2      """
3          MongoDB router for category, book, mobile, cloth apps
4      """
5
6      route_app_labels = [
7          "category",
8          "book",
9          "mobile",
10         "cloth",
11         "product",
12     ]
13
14     def db_for_read(self, model, **hints):
15         """
16             Attempts to read auth and contenttypes models go to auth_db.
17         """
18         if model._meta.app_label in self.route_app_labels:
19             return "ecom_mongodb"
20         return None
21
22     def db_for_write(self, model, **hints):
23         """
24             Attempts to write auth and contenttypes models go to auth_db.
25         """
26         if model._meta.app_label in self.route_app_labels:
27             return "ecom_mongodb"
28         return None
29
30     def allow_relation(self, obj1, obj2, **hints):
31         """
32             Allow relations if a model in the auth or contenttypes apps is
33         
```



```
1  class MongoRouter:
2      """
3          MongoDB router for category, book, mobile, cloth apps
4      """
5
6      route_app_labels = [
7          "category",
8          "book",
9          "mobile",
10         "cloth",
11         "product",
12     ]
13
14     def db_for_read(self, model, **hints):
15         """
16             Attempts to read auth and contenttypes models go to auth_db.
17         """
18         if model._meta.app_label in self.route_app_labels:
19             return "ecom_mongodb"
20         return None
21
22     def db_for_write(self, model, **hints):
23         """
24             Attempts to write auth and contenttypes models go to auth_db.
25         """
26         if model._meta.app_label in self.route_app_labels:
27             return "ecom_mongodb"
28         return None
29
30     def allow_relation(self, obj1, obj2, **hints):
31         """
32             Allow relations if a model in the auth or contenttypes apps is
33         
```



```

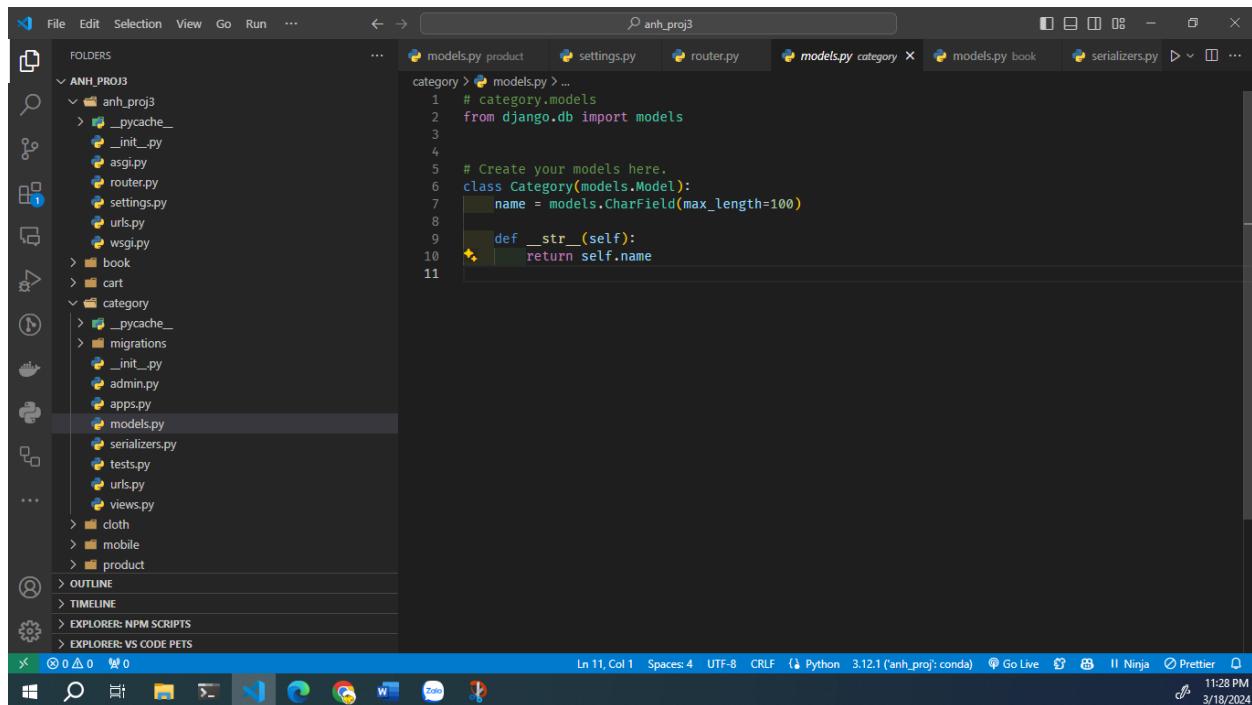
52     class MySQLRouter:
53         def db_for_read(self, model, **hints):
54             return 'ecom_mysql'
55
56         def db_for_write(self, model, **hints):
57             if model._meta.app_label in self.route_app_labels:
58                 return 'ecom_mysql'
59             return None
60
61         def allow_relation(self, obj1, obj2, **hints):
62             """
63                 Allow relations if a model in the auth or contenttypes apps is
64                 involved.
65             """
66             if (
67                 obj1._meta.app_label in self.route_app_labels
68                 or obj2._meta.app_label in self.route_app_labels
69             ):
70                 return True
71             return None
72
73         def allow_migrate(self, db, app_label, model_name=None, **hints):
74             if app_label in self.route_app_labels:
75                 return db == 'ecom_mysql'
76             return None
77
78
79
80
81
82
83
84
85
86
87

```

- Thêm chỉnh sửa CORS thành “\*” để tiện cho việc giao diện gọi API

- Category:

- Tạo model cho Category trong models.py

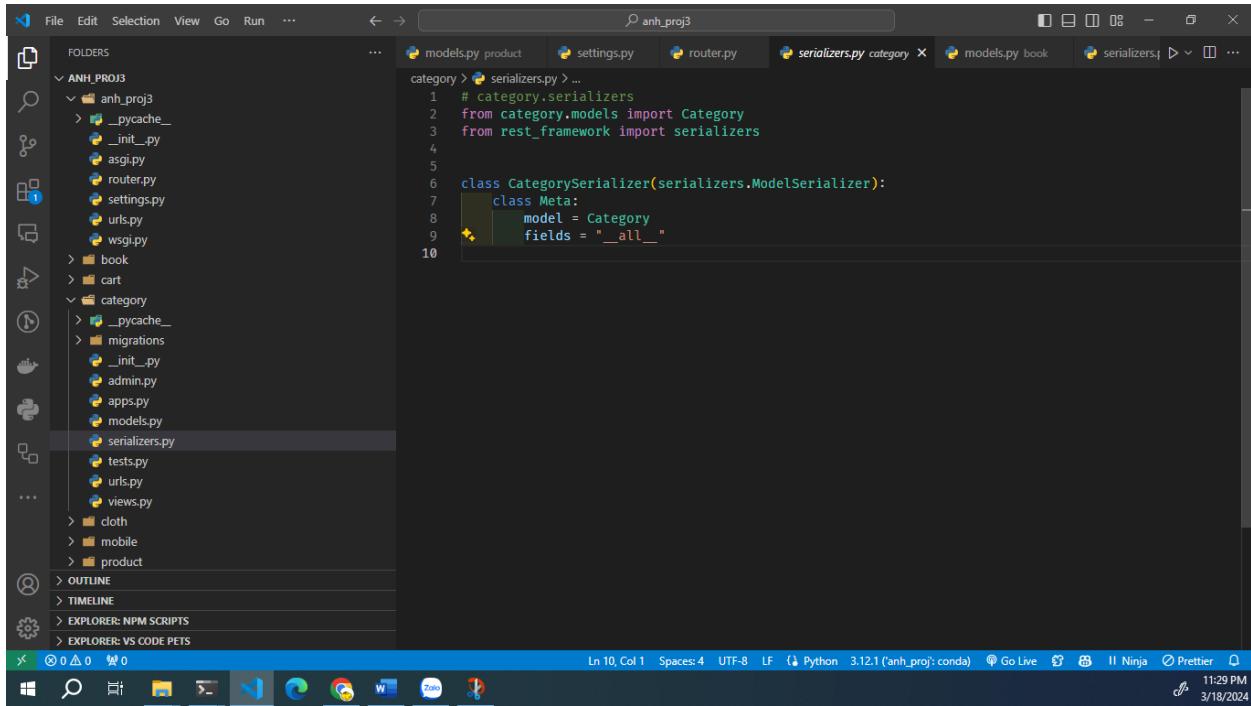


```

1  # category.models
2  from django.db import models
3
4
5  # Create your models here.
6  class Category(models.Model):
7      name = models.CharField(max_length=100)
8
9      def __str__(self):
10         return self.name
11

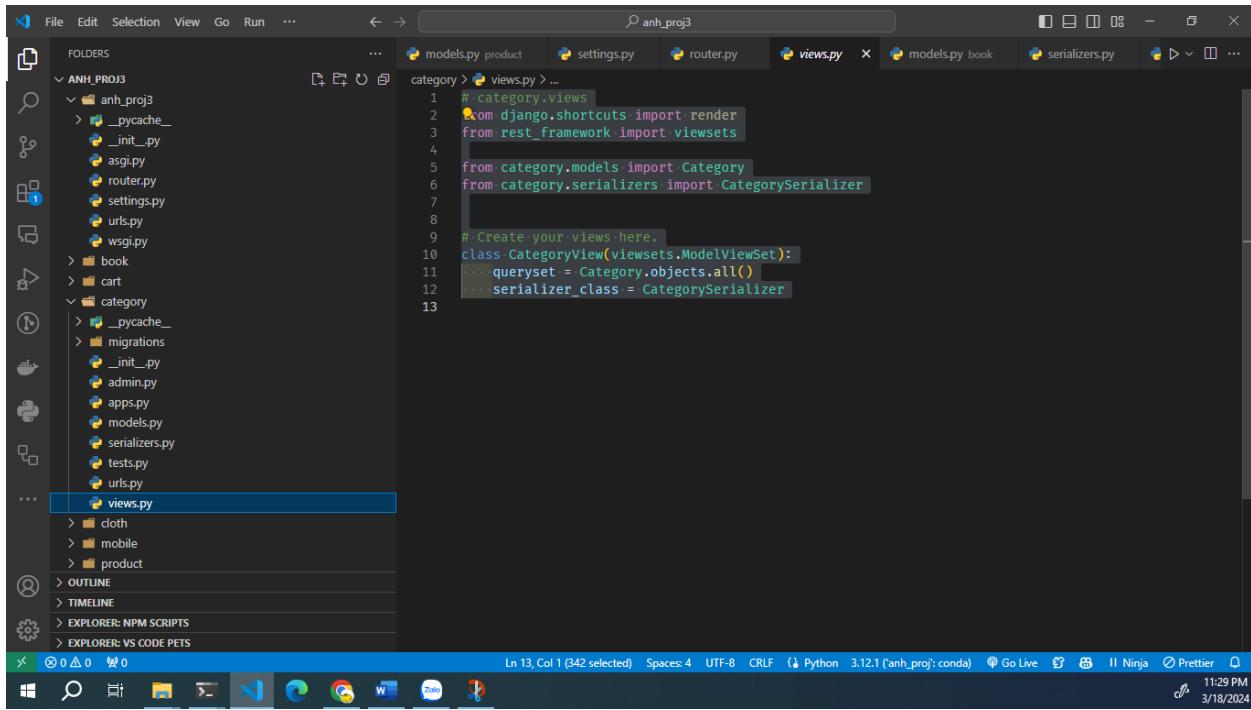
```

- Tạo serializers cho Category trong serializers.py



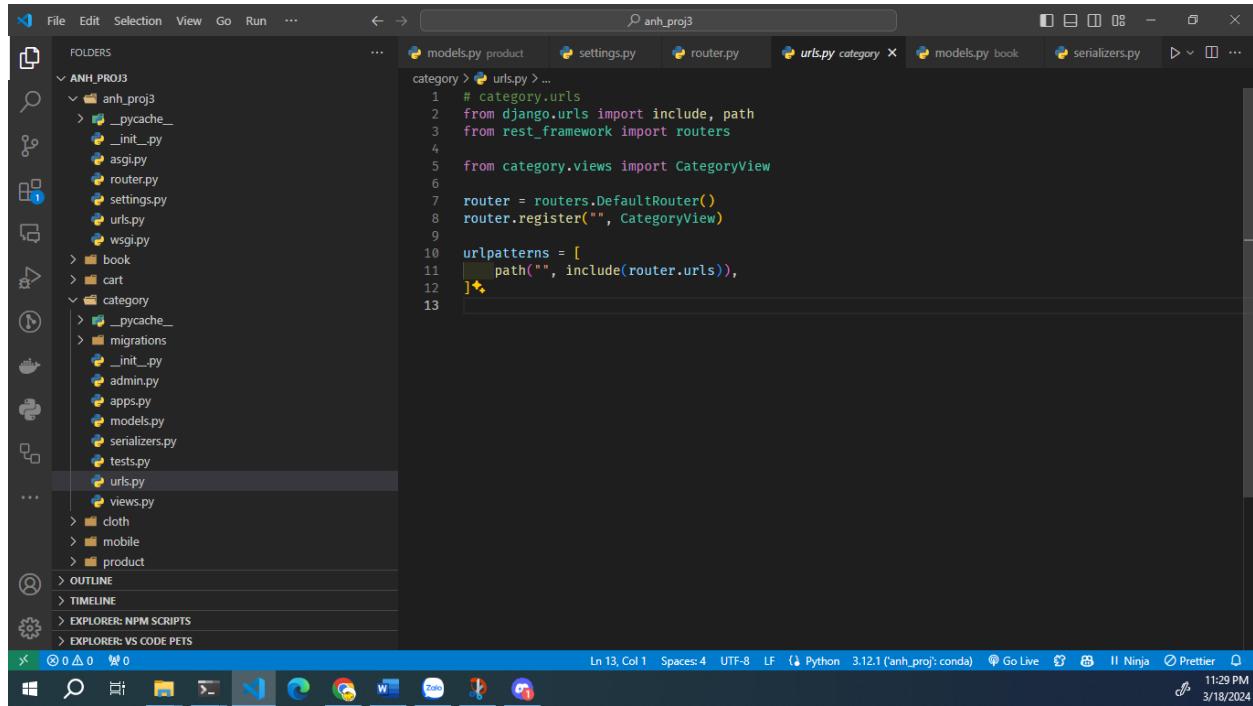
```
category > serializers.py > ...
1  # category.serializers
2  from category.models import Category
3  from rest_framework import serializers
4
5
6  class CategorySerializer(serializers.ModelSerializer):
7      model = Category
8      fields = "__all__"
9
10
```

- Tạo views cho category trong views.py



```
category > views.py > ...
1  # category.views
2  from django.shortcuts import render
3  from rest_framework import viewsets
4
5  from category.models import Category
6  from category.serializers import CategorySerializer
7
8
9  # Create your views here.
10 class CategoryView(viewsets.ModelViewSet):
11     queryset = Category.objects.all()
12     serializer_class = CategorySerializer
13
```

- Tạo urlpatterns cho category trong urls.py

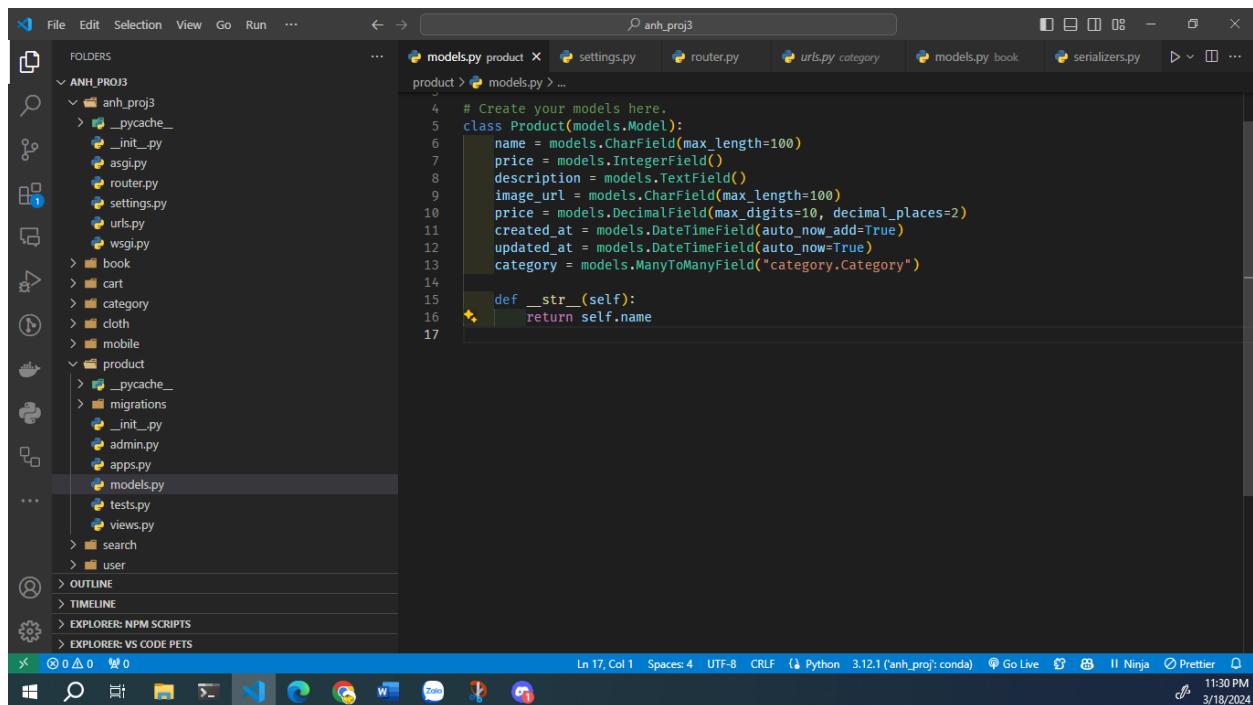


```

category > urls.py > ...
1  # category.urls
2  from django.urls import include, path
3  from rest_framework import routers
4
5  from category.views import CategoryView
6
7  router = routers.DefaultRouter()
8  router.register("", CategoryView)
9
10 urlpatterns = [
11     path("", include(router.urls)),
12 ]
13

```

- Product:
  - Tạo model cho Product trong models.py

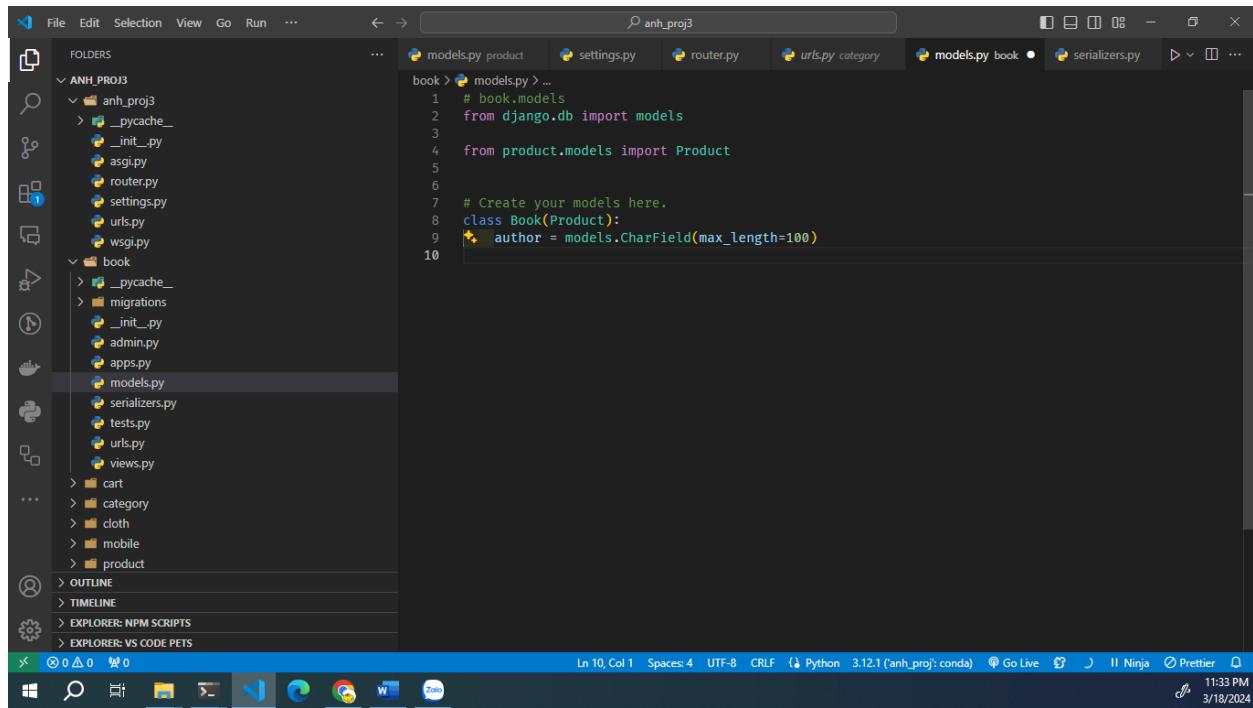


```

product > models.py > ...
4  # Create your models here.
5  class Product(models.Model):
6      name = models.CharField(max_length=100)
7      price = models.IntegerField()
8      description = models.TextField()
9      image_url = models.CharField(max_length=100)
10     price = models.DecimalField(max_digits=10, decimal_places=2)
11     created_at = models.DateTimeField(auto_now_add=True)
12     updated_at = models.DateTimeField(auto_now=True)
13     category = models.ManyToManyField("category.Category")
14
15     def __str__(self):
16         return self.name
17

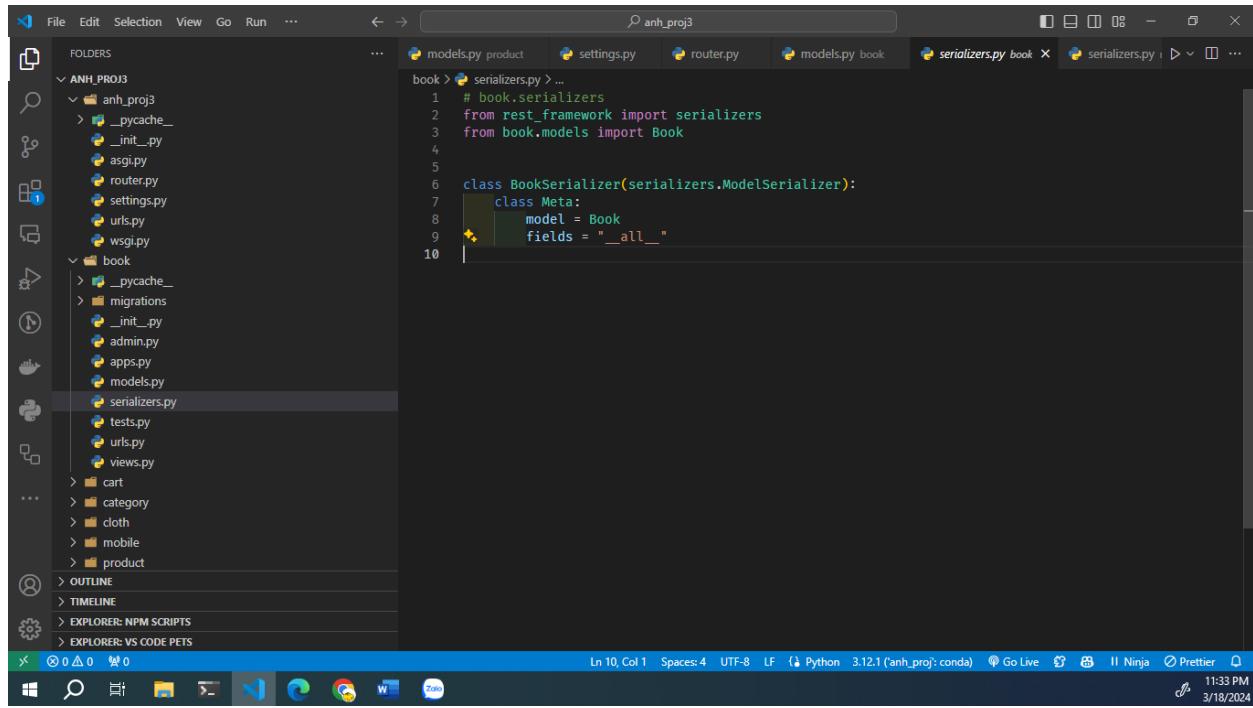
```

- Book:
  - Tạo model cho book extends từ product trong models.py



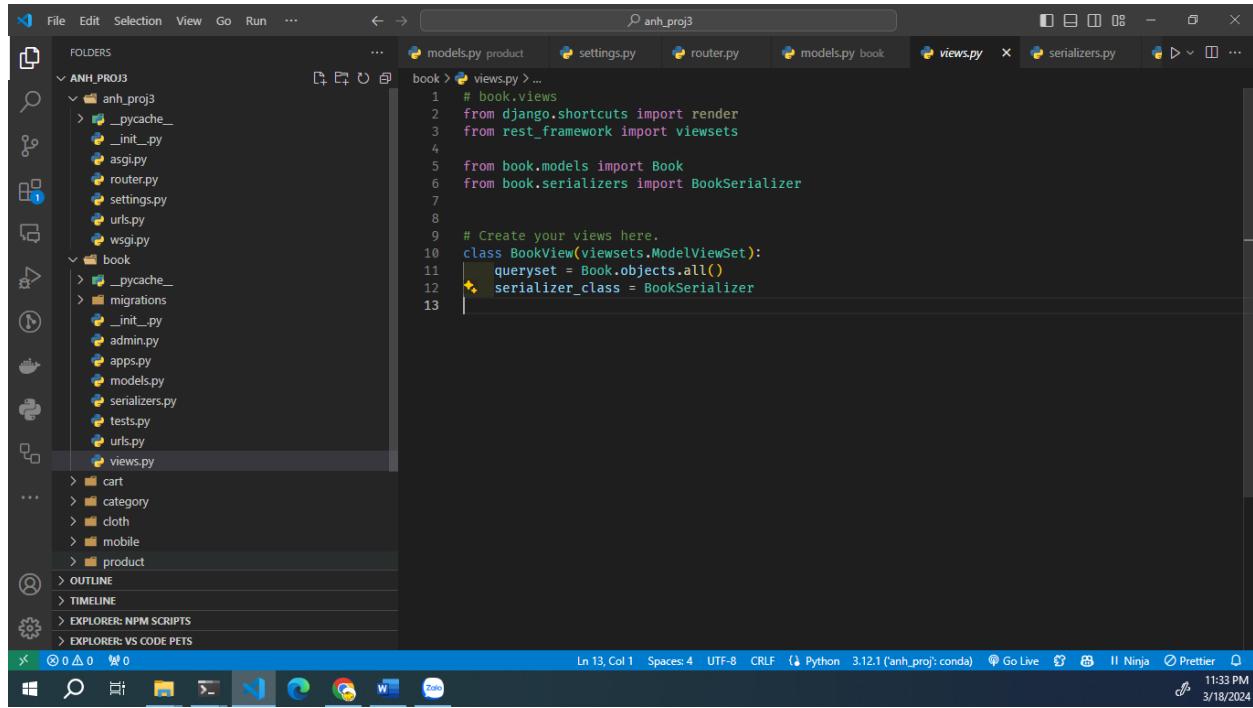
```
book > models.py > ...
1  # book.models
2  from django.db import models
3
4  from product.models import Product
5
6
7  # Create your models here.
8  class Book(Product):
9      author = models.CharField(max_length=100)
10
```

○ Tạo serializers cho book trong serializers.py



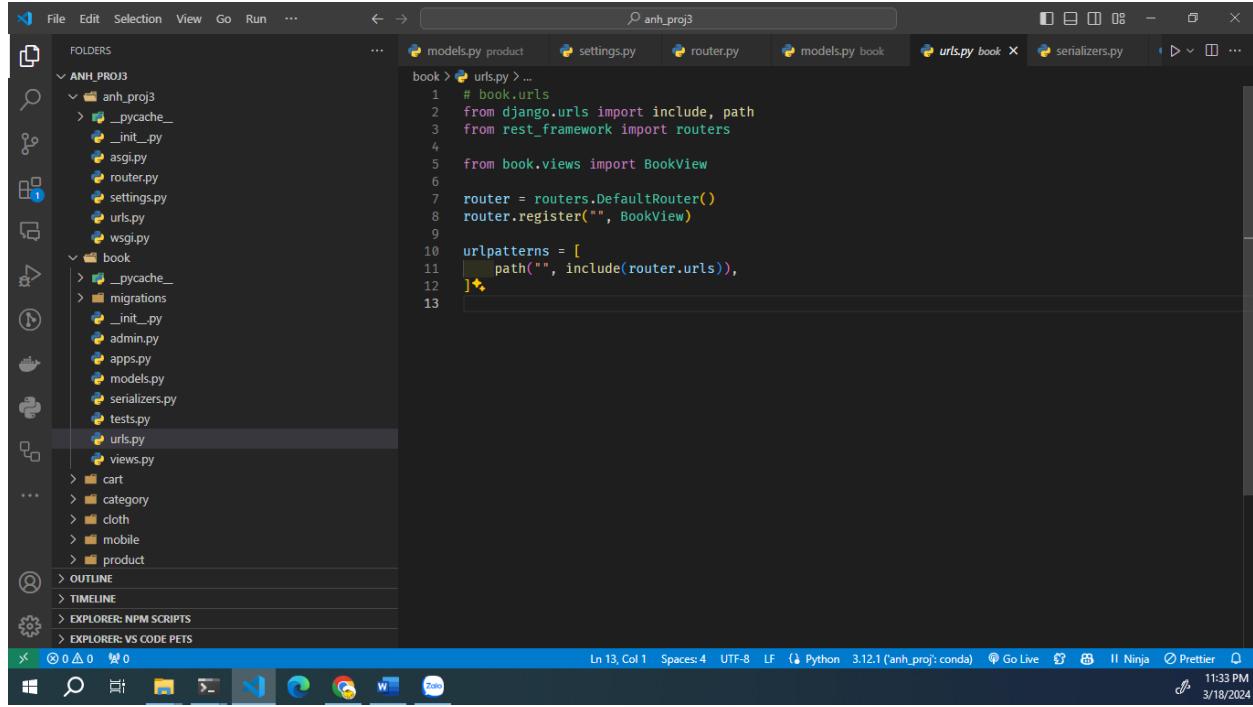
```
book > serializers.py > ...
1  # book.serializers
2  from rest_framework import serializers
3  from book.models import Book
4
5
6  class BookSerializer(serializers.ModelSerializer):
7      class Meta:
8          model = Book
9          fields = "__all__"
10
```

○ Tạo views cho book trong views.py



```
book > views.py > ...
1  # book.views
2  from django.shortcuts import render
3  from rest_framework import viewsets
4
5  from book.models import Book
6  from book.serializers import BookSerializer
7
8
9  # Create your views here.
10 class BookView(viewsets.ModelViewSet):
11     queryset = Book.objects.all()
12     serializer_class = BookSerializer
```

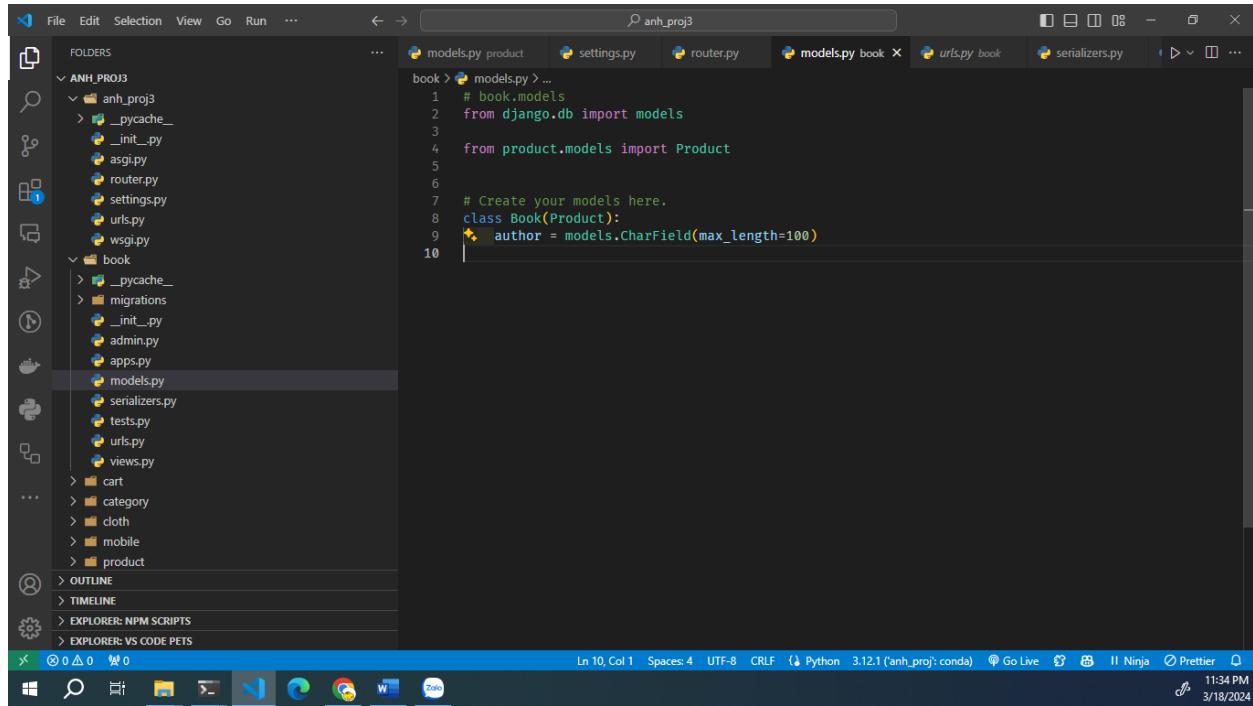
- Tạo urlpatterns cho book trong urls.py



```
book > urls.py > ...
1  # book.urls
2  from django.urls import include, path
3  from rest_framework import routers
4
5  from book.views import BookView
6
7  router = routers.DefaultRouter()
8  router.register("", BookView)
9
10 urlpatterns = [
11     path("", include(router.urls)),
12 ]
```

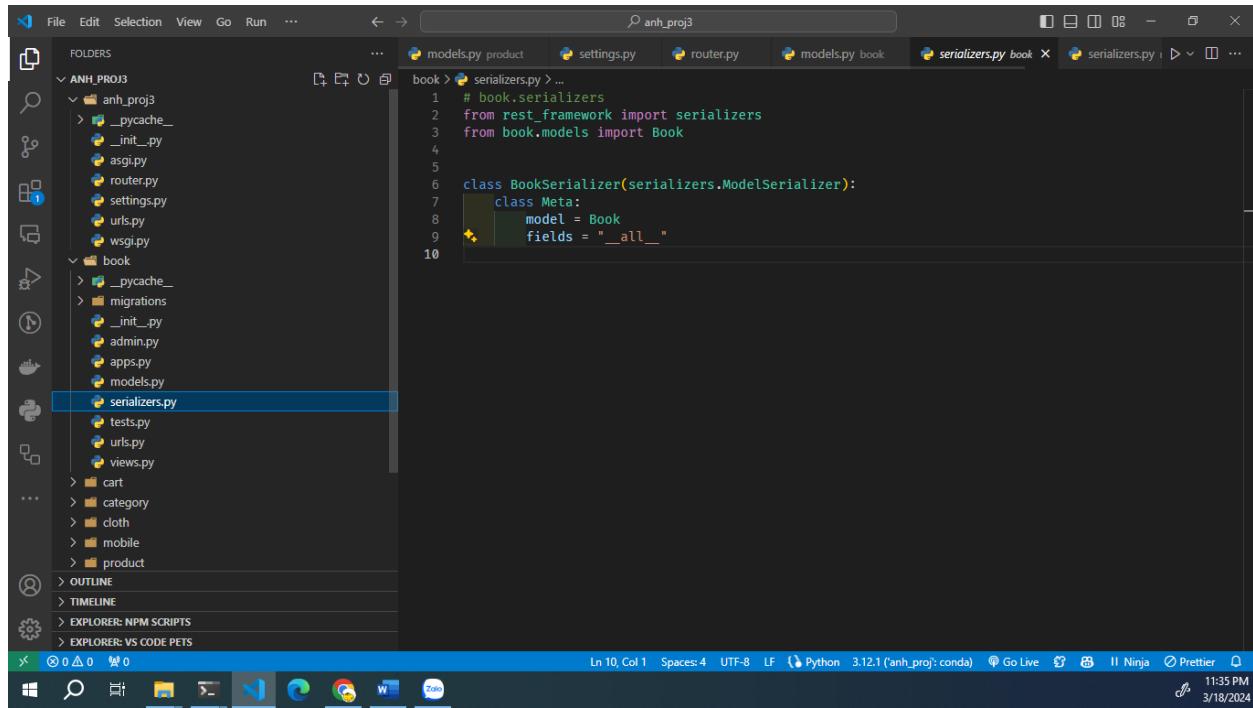
- Mobile:

- Tạo model cho mobile extends từ product trong models.py



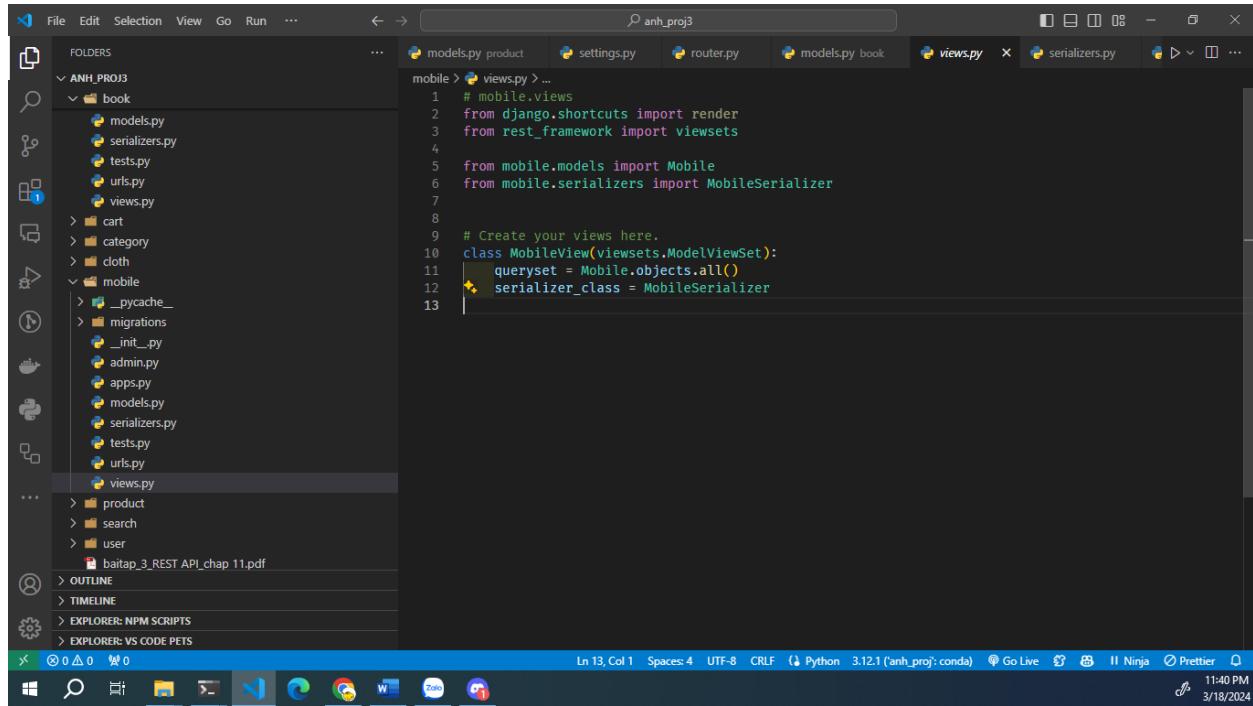
```
book > models.py > ...
1  # book.models
2  from django.db import models
3
4  from product.models import Product
5
6
7  # Create your models here.
8  class Book(Product):
9      author = models.CharField(max_length=100)
10
```

- Tạo serializers cho mobile trong serializers.py



```
book > serializers.py > ...
1  # book.serializers
2  from rest_framework import serializers
3  from book.models import Book
4
5
6  class BookSerializer(serializers.ModelSerializer):
7      class Meta:
8          model = Book
9          fields = "__all__"
10
```

- Tạo views cho mobile trong views.py

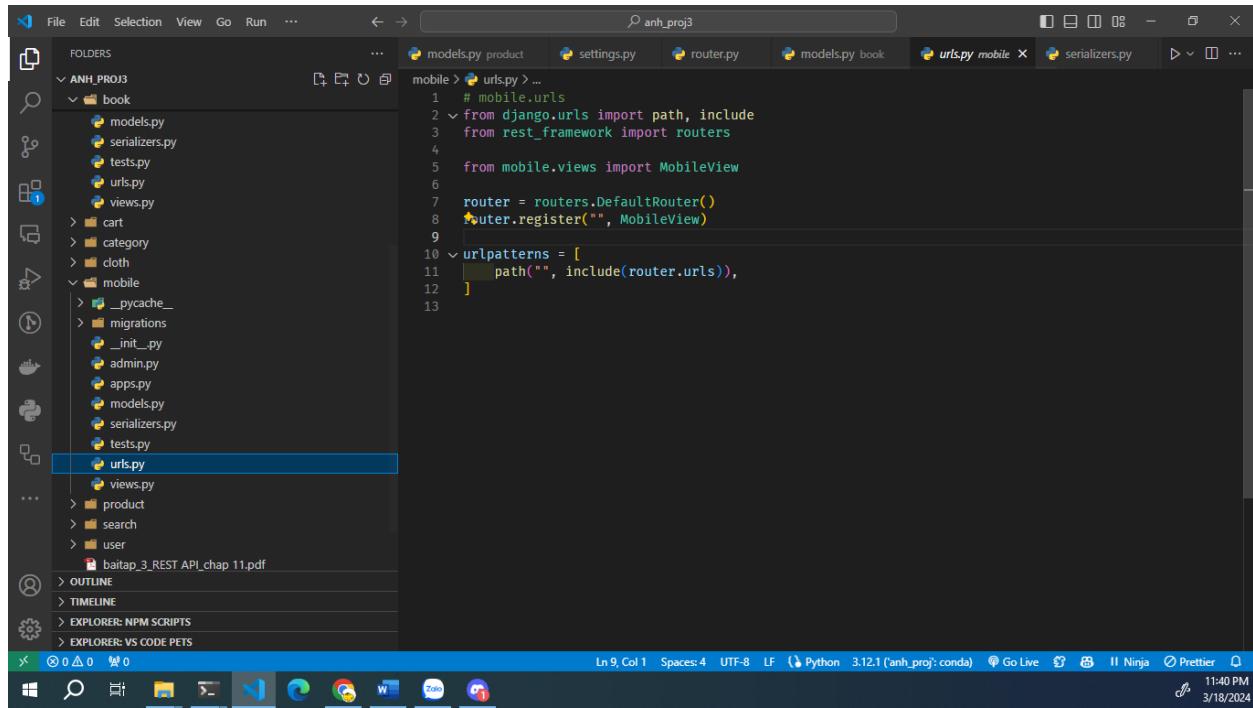


```

mobile > views.py > ...
1  # mobile.views
2  from django.shortcuts import render
3  from rest_framework import viewsets
4
5  from mobile.models import Mobile
6  from mobile.serializers import MobileSerializer
7
8
9  # Create your views here.
10 class MobileView(viewsets.ModelViewSet):
11     queryset = Mobile.objects.all()
12     serializer_class = MobileSerializer
13

```

- Tạo urlpatterns cho mobile trong urls.py



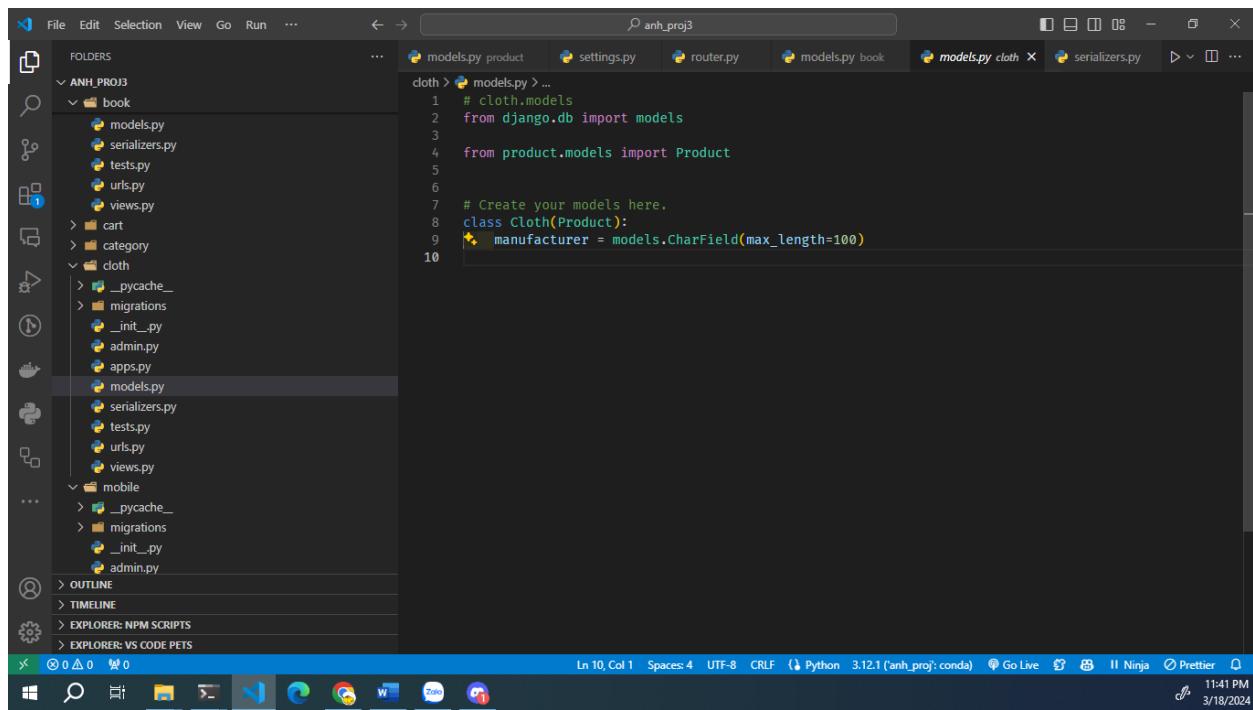
```

mobile > urls.py > ...
1  # mobile.urls
2  from django.urls import path, include
3  from rest_framework import routers
4
5  from mobile.views import MobileView
6
7  router = routers.DefaultRouter()
8  router.register("", MobileView)
9
10 urlpatterns = [
11     path("", include(router.urls)),
12 ]
13

```

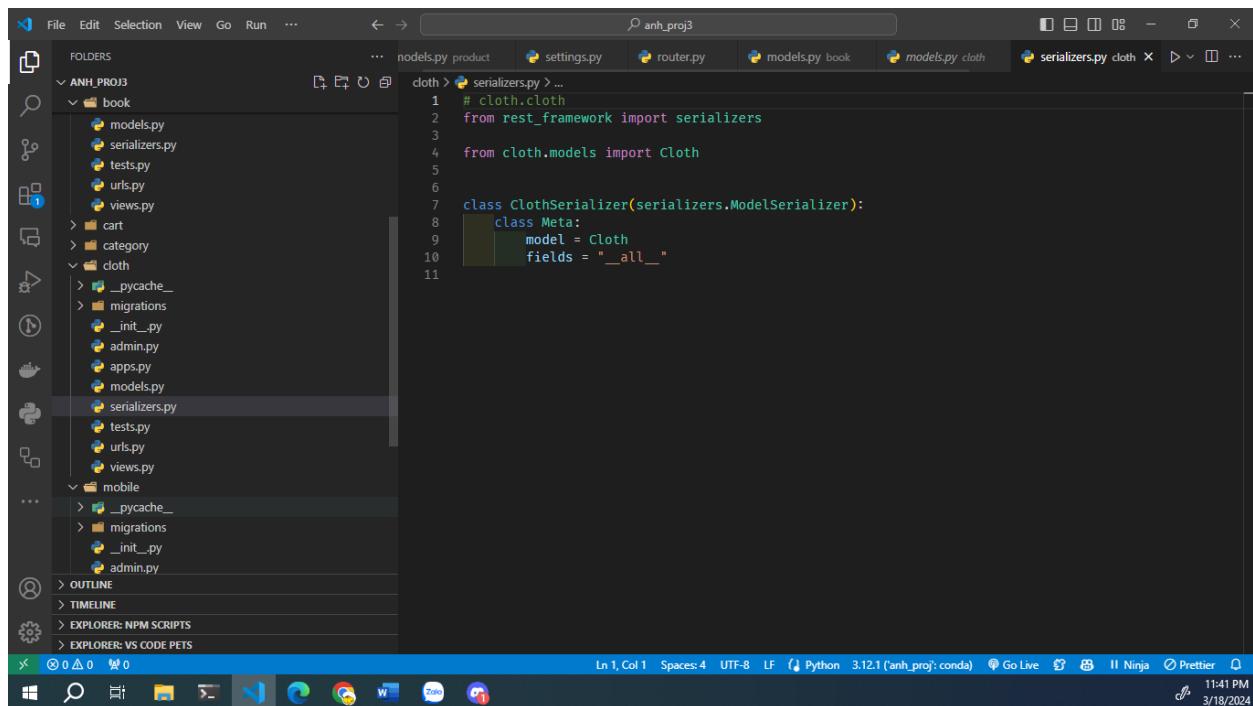
- Cloth:

- Tạo model cho cloth trong models.py



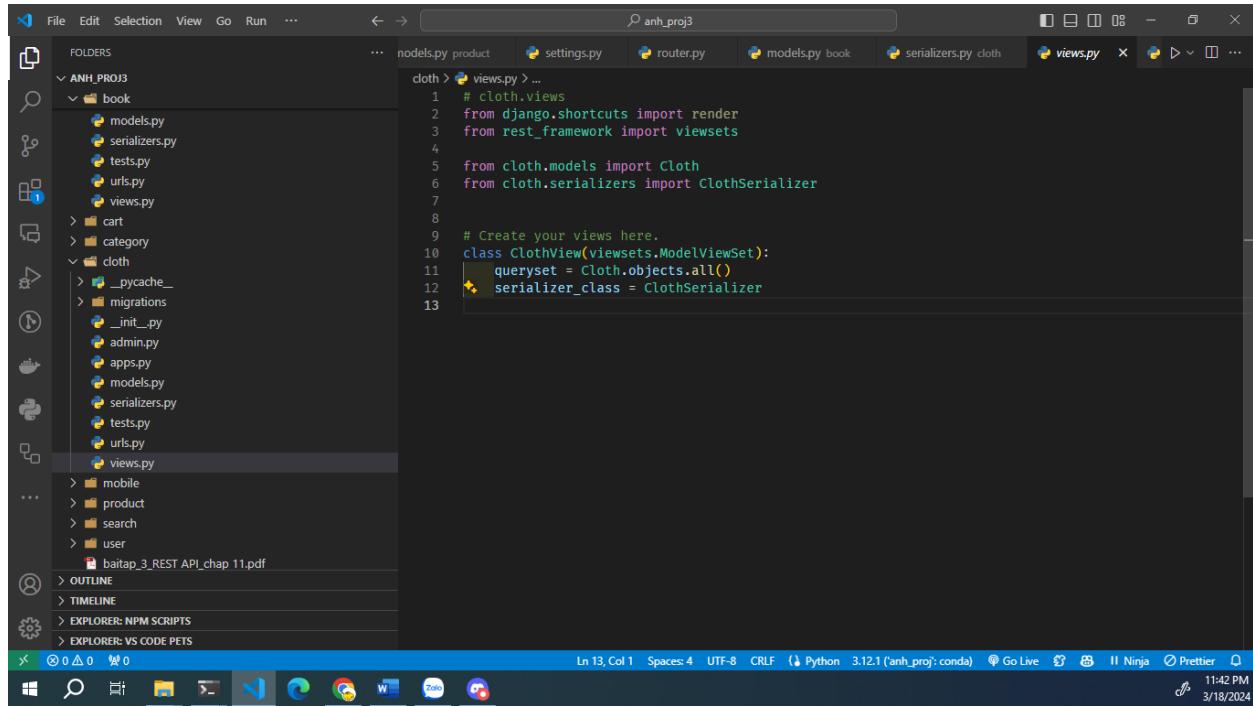
```
1 # cloth.models
2 from django.db import models
3
4 from product.models import Product
5
6
7 # Create your models here.
8 class Cloth(Product):
9     manufacturer = models.CharField(max_length=100)
10
```

- Tạo serializers cho cloth trong serializers.py



```
1 # cloth.serializers
2 from rest_framework import serializers
3
4 from cloth.models import Cloth
5
6
7 class ClothSerializer(serializers.ModelSerializer):
8     model = Cloth
9     fields = "__all__"
10
```

- Tạo views cho cloth trong views.py



```

File Edit Selection View Go Run ...
FOLDERS
ANH_PROJ
book
models.py
serializers.py
tests.py
urls.py
views.py
> cart
> category
cloth
> __pycache__
> migrations
__init__.py
admin.py
apps.py
models.py
serializers.py
tests.py
urls.py
views.py
> mobile
> product
> search
> user
baitap_3_REST API_chap 11.pdf
OUTLINE
TIMELINE
EXPLORER: NPM SCRIPTS
EXPLORER: VS CODE PETS
Ln 13, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.1 (anh_proj):conda Go Live II Ninja Prettier
11:42 PM 3/18/2024

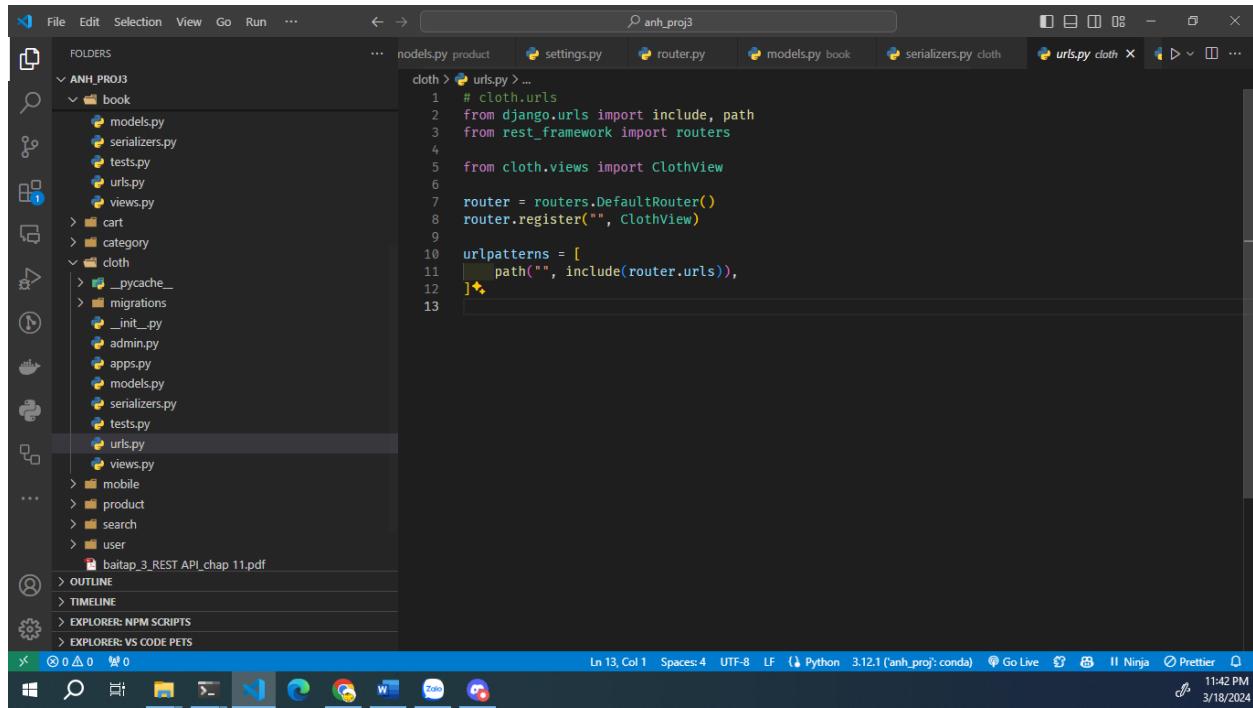
```

```

cloth > views.py > ...
1 # cloth.views
2 from django.shortcuts import render
3 from rest_framework import viewsets
4
5 from cloth.models import Cloth
6 from cloth.serializers import ClothSerializer
7
8
9 # Create your views here.
10 class ClothViewSet(viewsets.ModelViewSet):
11     queryset = Cloth.objects.all()
12     serializer_class = ClothSerializer
13

```

- Tạo urlpatterns cho cloth trong urls.py



```

File Edit Selection View Go Run ...
FOLDERS
ANH_PROJ
book
models.py
serializers.py
tests.py
urls.py
views.py
> cart
> category
cloth
> __pycache__
> migrations
__init__.py
admin.py
apps.py
models.py
serializers.py
tests.py
urls.py
views.py
> mobile
> product
> search
> user
baitap_3_REST API_chap 11.pdf
OUTLINE
TIMELINE
EXPLORER: NPM SCRIPTS
EXPLORER: VS CODE PETS
Ln 13, Col 1 Spaces: 4 UTF-8 LF Python 3.12.1 (anh_proj):conda Go Live II Ninja Prettier
11:42 PM 3/18/2024

```

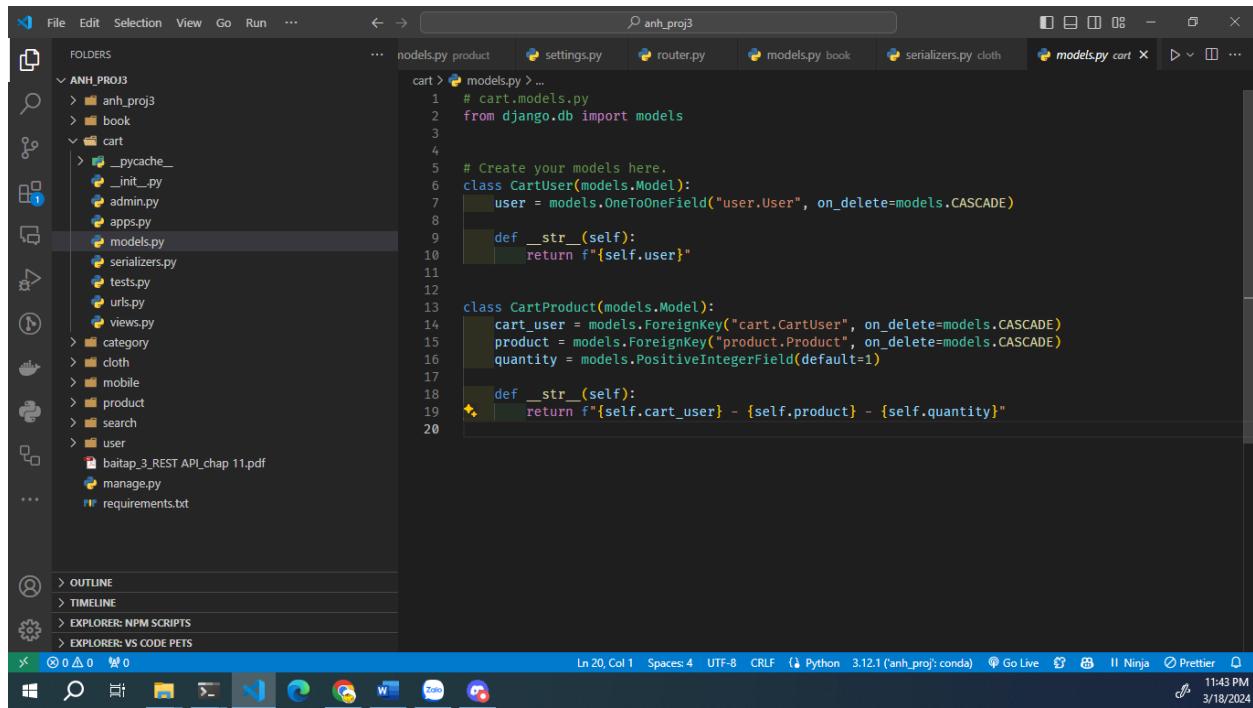
```

cloth > urls.py > ...
1 # cloth.urls
2 from django.urls import include, path
3 from rest_framework import routers
4
5 from cloth.views import ClothViewSet
6
7 router = routers.DefaultRouter()
8 router.register("", ClothViewSet)
9
10 urlpatterns = [
11     path("", include(router.urls)),
12 ]
13

```

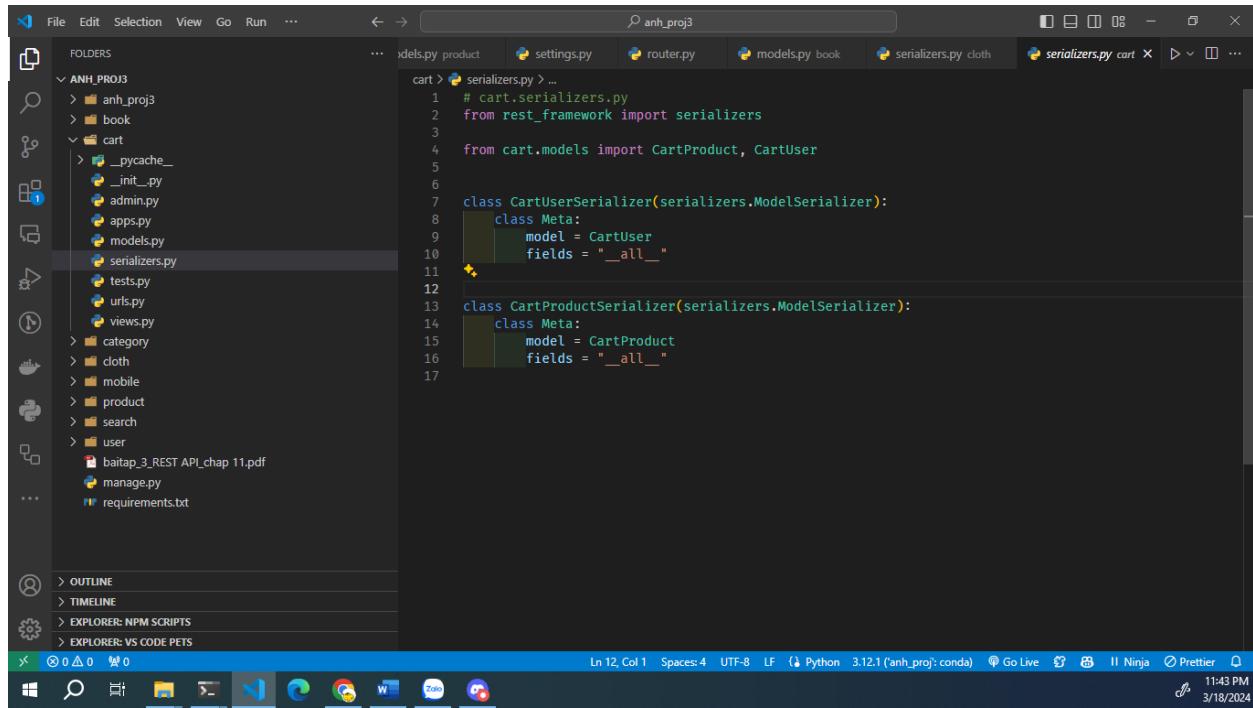
- Cart:

- Tạo model cho cart trong models.py



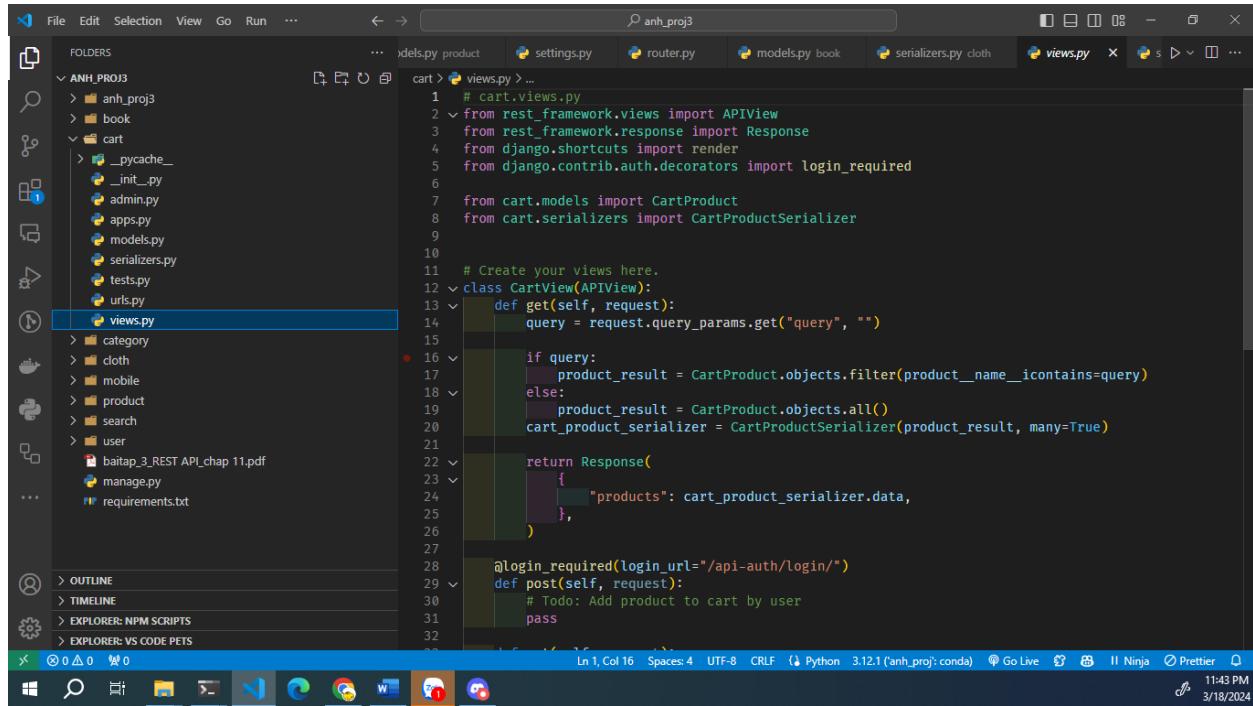
```
cart > models.py > ...
1  # cart.models.py
2  from django.db import models
3
4
5  # Create your models here.
6  class CartUser(models.Model):
7      user = models.OneToOneField("user.User", on_delete=models.CASCADE)
8
9      def __str__(self):
10         return f"{self.user}"
11
12
13  class CartProduct(models.Model):
14      cart_user = models.ForeignKey("cart.CartUser", on_delete=models.CASCADE)
15      product = models.ForeignKey("product.Product", on_delete=models.CASCADE)
16      quantity = models.PositiveIntegerField(default=1)
17
18      def __str__(self):
19         return f"{self.cart_user} - {self.product} - {self.quantity}"
20
```

○ Tạo serializers cho cart trong serializers.py



```
cart > serializers.py > ...
1  # cart.serializers.py
2  from rest_framework import serializers
3
4  from cart.models import CartProduct, CartUser
5
6
7  class CartUserSerializer(serializers.ModelSerializer):
8      class Meta:
9          model = CartUser
10         fields = "__all__"
11
12
13  class CartProductSerializer(serializers.ModelSerializer):
14      class Meta:
15          model = CartProduct
16          fields = "__all__"
17
```

○ Tạo views cho cart trong views.py

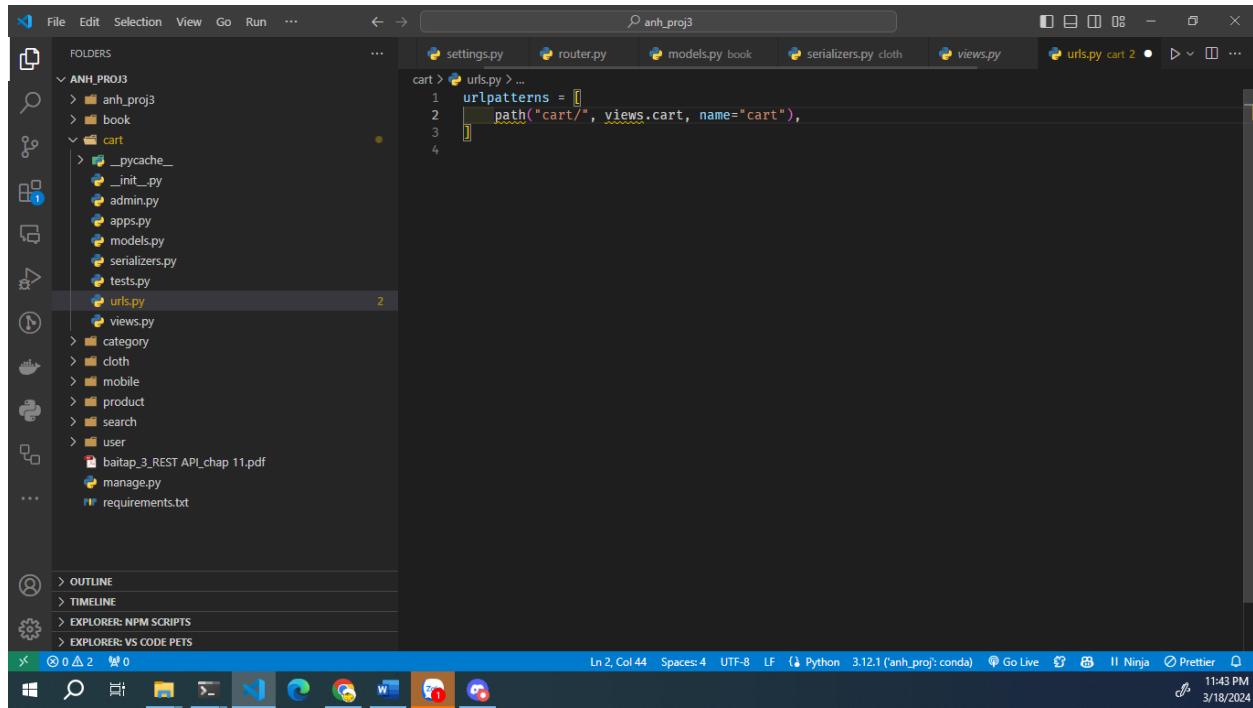


```

1  # cart.views.py
2  from rest_framework.views import APIView
3  from rest_framework.response import Response
4  from django.shortcuts import render
5  from django.contrib.auth.decorators import login_required
6
7  from cart.models import CartProduct
8  from cart.serializers import CartProductSerializer
9
10
11 # Create your views here.
12 class CartView(APIView):
13     def get(self, request):
14         query = request.query_params.get("query", "")
15
16         if query:
17             product_result = CartProduct.objects.filter(product__name__icontains=query)
18         else:
19             product_result = CartProduct.objects.all()
20         cart_product_serializer = CartProductSerializer(product_result, many=True)
21
22         return Response(
23             {
24                 "products": cart_product_serializer.data,
25             }
26         )
27
28     @login_required(login_url="/api-auth/login/")
29     def post(self, request):
30         # Todo: Add product to cart by user
31         pass
32

```

- Tạo urlpatterns cho cart trong urls.py

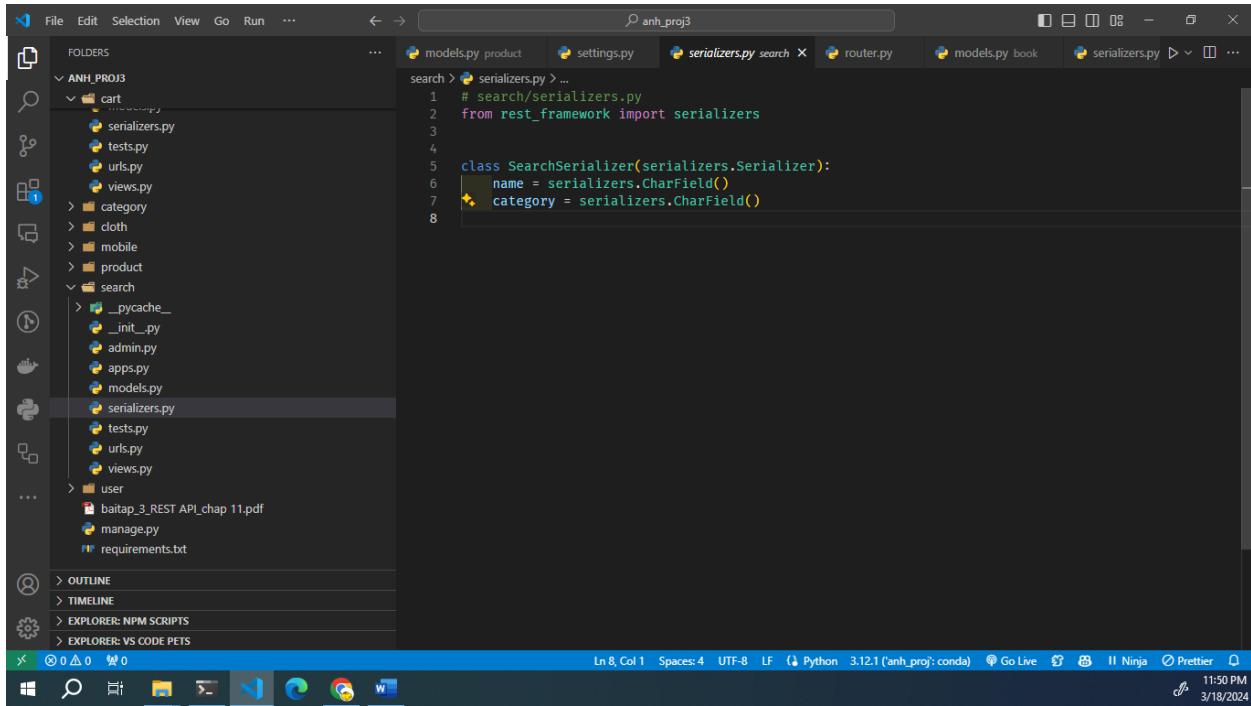


```

1  urlpatterns = [
2      path("cart/", views.cart, name="cart"),
3  ]

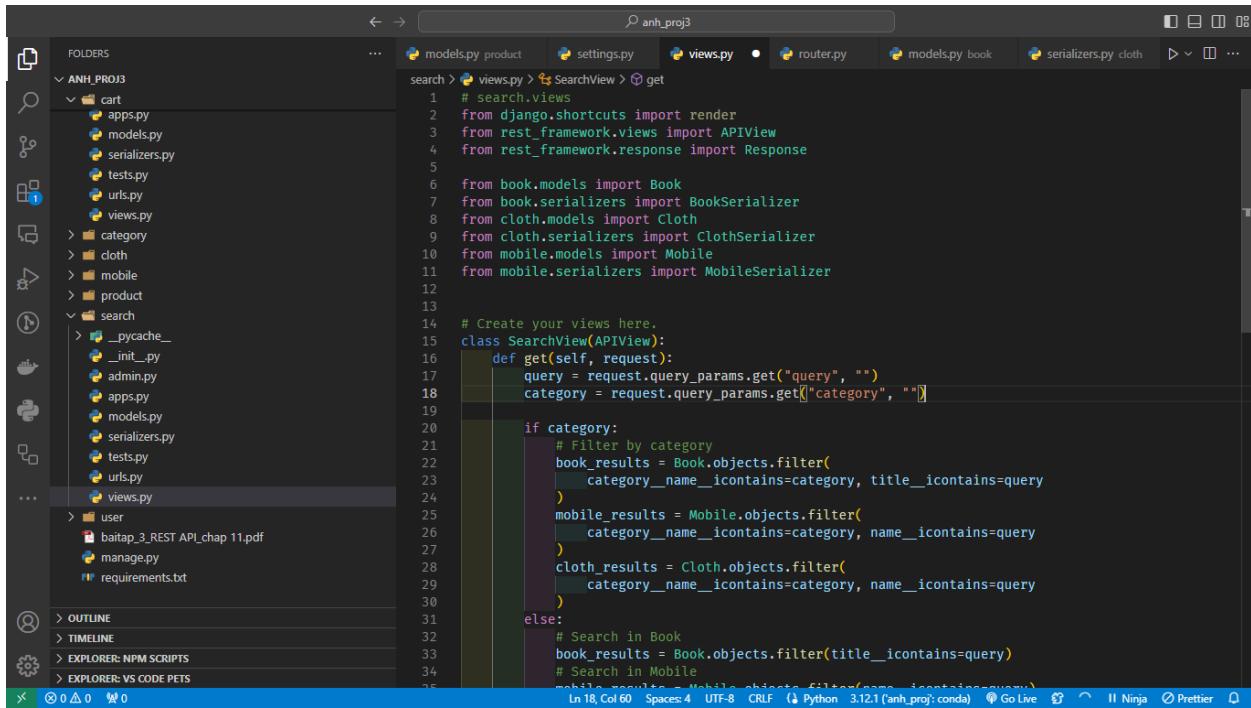
```

- Search:
  - Tạo serializers cho search



```
search > serializers.py > ...
1  # search/serializers.py
2  from rest_framework import serializers
3
4
5  class SearchSerializer(serializers.Serializer):
6      name = serializers.CharField()
7      category = serializers.CharField()
```

## ○ Tạo views cho search



```
search > views.py > SearchView > get
1  # search/views.py
2  from django.shortcuts import render
3  from rest_framework.views import APIView
4  from rest_framework.response import Response
5
6  from book.models import Book
7  from book.serializers import BookSerializer
8  from cloth.models import Cloth
9  from cloth.serializers import ClothSerializer
10 from mobile.models import Mobile
11 from mobile.serializers import MobileSerializer
12
13
14 # Create your views here.
15 class SearchView(APIView):
16     def get(self, request):
17         query = request.query_params.get("query", "")
18         category = request.query_params.get("category", "")
19
20         if category:
21             # Filter by category
22             book_results = Book.objects.filter(
23                 category__name__icontains=category, title__icontains=query
24             )
25             mobile_results = Mobile.objects.filter(
26                 category__name__icontains=category, name__icontains=query
27             )
28             cloth_results = Cloth.objects.filter(
29                 category__name__icontains=category, name__icontains=query
30             )
31         else:
32             # Search in Book
33             book_results = Book.objects.filter(title__icontains=query)
34             # Search in Mobile
35             mobile_results = Mobile.objects.filter(name__icontains=query)
```

```

15     class SearchView(APIView):
16         def get(self, request):
17             if category__name__icontains:=category, name__icontains=query
18                 )
19             else:
20                 # Search in Book
21                 book_results = Book.objects.filter(title__icontains=query)
22                 # Search in Mobile
23                 mobile_results = Mobile.objects.filter(name__icontains=query)
24                 # Search in Cloth
25                 cloth_results = Cloth.objects.filter(name__icontains=query)
26
27             # Serialize results
28             book_serializer = BookSerializer(book_results, many=True)
29             mobile_serializer = MobileSerializer(mobile_results, many=True)
30             cloth_serializer = ClothSerializer(cloth_results, many=True)
31
32             return Response(
33                 {
34                     "books": book_serializer.data,
35                     "mobiles": mobile_serializer.data,
36                     "cloths": cloth_serializer.data,
37                 }
38             )
39
40
41
42
43
44
45
46
47
48
49
50
51

```

- Tạo urlpatterns cho search

```

1 # search.urls
2 from django.urls import path
3 from search.views import SearchView
4
5 urlpatterns = [
6     path("", SearchView.as_view(), name="search"),
7 ]

```

- User:

- Tạo serializers cho user

```

1  # serializers.py
2  from rest_framework import serializers
3  from django.contrib.auth.models import User
4
5
6  class RegisterSerializer(serializers.ModelSerializer):
7      class Meta:
8          model = User
9          fields = ["username", "email", "password"]
10         extra_kwargs = {"password": {"write_only": True}}
11
12
13  class UserSerializer(serializers.ModelSerializer):
14      class Meta:
15          model = User
16          fields = "__all__"
17

```

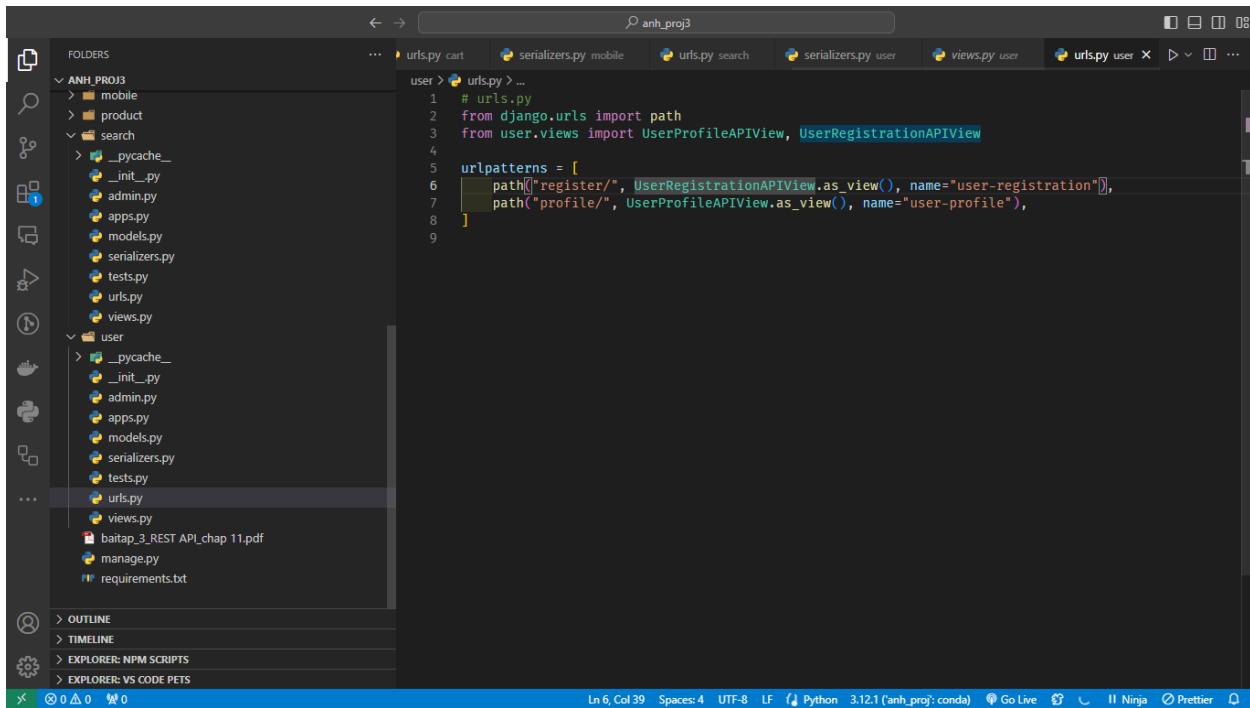
## ○ Tạo views cho user

```

4  from rest_framework.views import APIView
5  from django.contrib.auth.models import User
6  from user.serializers import RegisterSerializer, UserSerializer
7
8
9  class UserRegistrationAPIView(APIView):
10     def post(self, request):
11         serializer = RegisterSerializer(data=request.data)
12         if serializer.is_valid():
13             username = serializer.validated_data["username"]
14             email = serializer.validated_data["email"]
15             password = serializer.validated_data["password"]
16
17             # Create a new user instance
18             new_user = User.objects.create_user(
19                 username=username, email=email, password=password
20             )
21
22             return Response(serializer.data, status=status.HTTP_201_CREATED)
23         return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
24
25
26 class UserProfileAPIView(APIView):
27     permission_classes = [permissions.IsAuthenticated]
28
29     def get(self, request):
30         serializer = UserSerializer(request.user)
31         return Response(serializer.data)
32

```

## ○ Tạo urlpatterns cho user



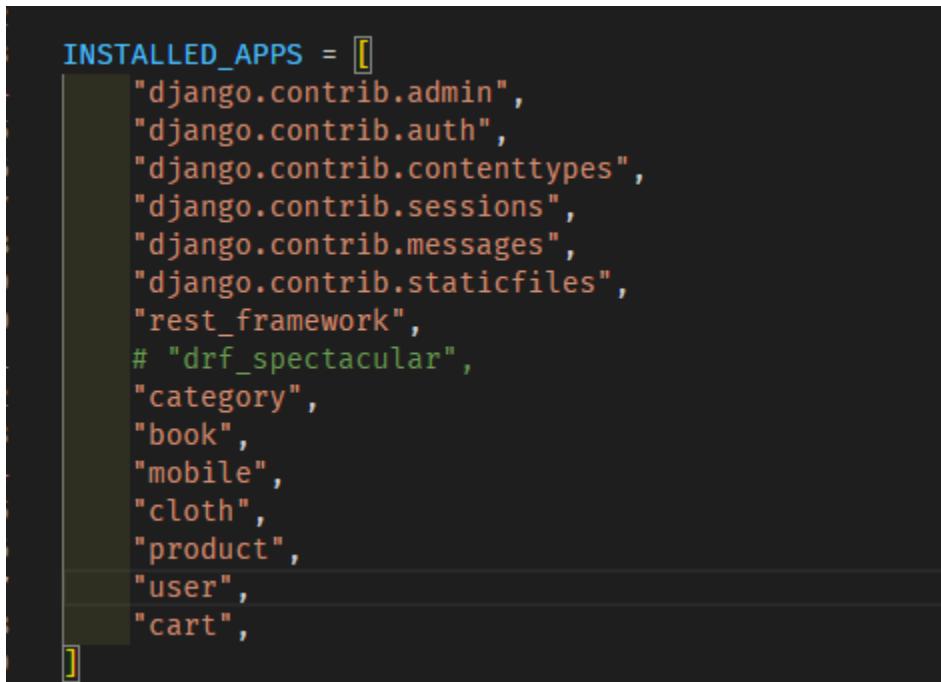
The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under 'anh\_proj3'. It includes 'ANH\_PROJ3' (containing 'mobile', 'product', 'search', and 'user' apps), 'urls.py' (containing 'cart', 'mobile', 'search', and 'user' imports), and 'user' (containing '\_pycache\_ files, 'admin.py', 'apps.py', 'models.py', 'serializers.py', 'tests.py', 'urls.py', and 'views.py').
- Code Editor:** The 'urls.py' file in the 'user' app is open, showing the following code:

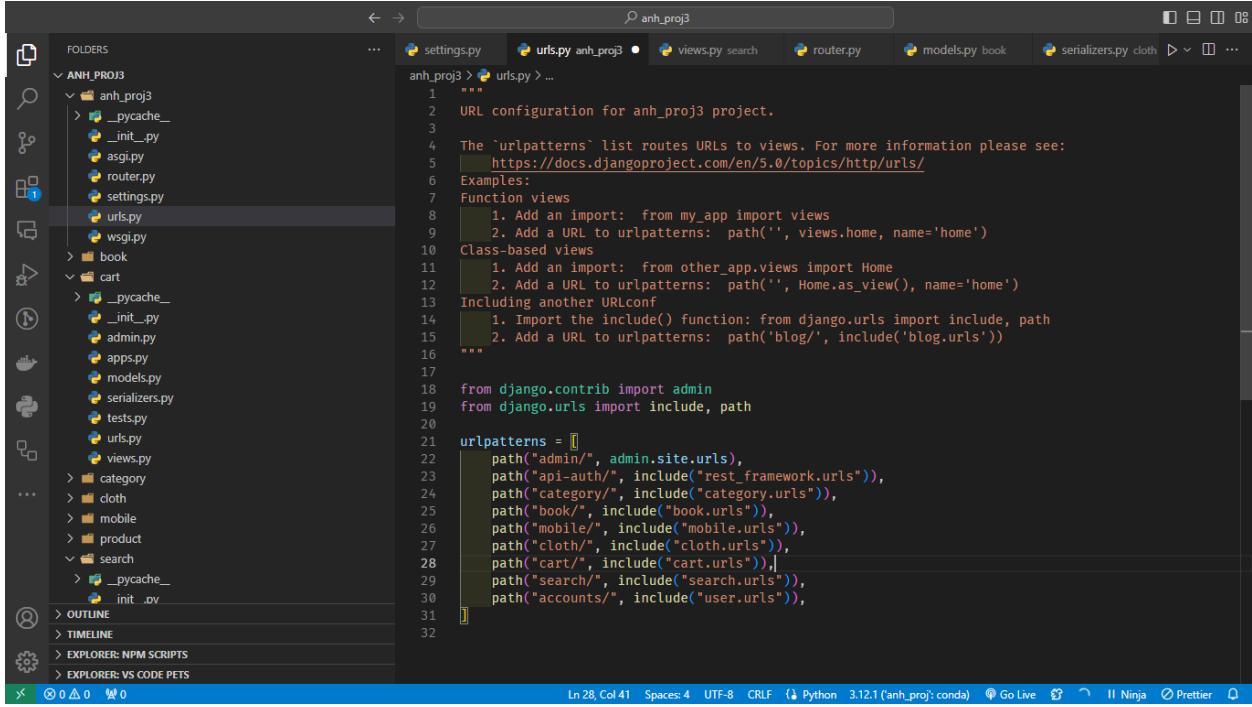
```
1 # urls.py
2 from django.urls import path
3 from user.views import UserRegistrationAPIView, UserProfileAPIView
4
5 urlpatterns = [
6     path("register/", UserRegistrationAPIView.as_view(), name="user-registration"),
7     path("profile/", UserProfileAPIView.as_view(), name="user-profile"),
8 ]
```

- Bottom Status Bar:** Shows 'Ln 6, Col 39', 'Spaces: 4', 'UTF-8', 'LF', 'Python 3.12.1 ('anh\_proj3':conda)', 'Go Live', 'Ninja', 'Prettier', and other icons.

- Register các app trên vào INSTALLED\_APPS và urls.py ở root



```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "rest_framework",
    # "drf_spectacular",
    "category",
    "book",
    "mobile",
    "cloth",
    "product",
    "user",
    "cart",
]
```



The screenshot shows the VS Code interface with the 'anh\_proj3' project open. The left sidebar displays the project structure, including the 'anh\_proj3' folder containing 'book', 'cart', 'category', 'cloth', 'mobile', 'product', 'search', and 'user' subfolders, along with 'urls.py', 'settings.py', and 'wsgi.py'. The right panel shows the 'urls.py' file content:

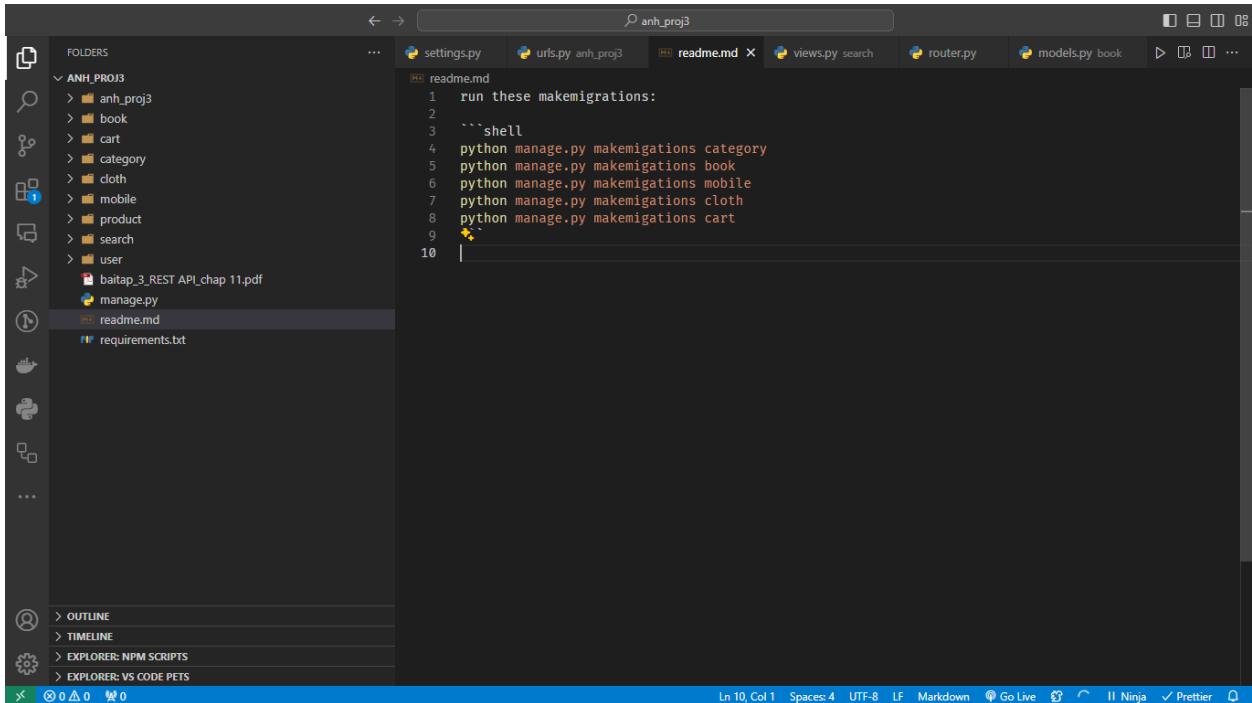
```

anh_proj3 > urls.py > ...
1 """
2 URL configuration for anh_proj3 project.
3
4 The `urlpatterns` list routes URLs to views. For more information please see:
5     https://docs.djangoproject.com/en/5.0/topics/http/urls/
6 Examples:
7 Function based views
8     1. Add an import: from my_app import views
9         2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11     1. Add an import: from other_app.views import Home
12         2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14     1. Import the include() function: from django.urls import include, path
15         2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17
18 from django.contrib import admin
19 from django.urls import include, path
20
21 urlpatterns = [
22     path("admin/", admin.site.urls),
23     path("api-auth/", include("rest_framework.urls")),
24     path("category/", include("category.urls")),
25     path("book/", include("book.urls")),
26     path("mobile/", include("mobile.urls")),
27     path("cloth/", include("cloth.urls")),
28     path("cart/", include("cart.urls")),
29     path("search/", include("search.urls")),
30     path("accounts/", include("user.urls")),
31 ]

```

The status bar at the bottom indicates: Ln 28, Col 41, Spaces: 4, UTF-8, CRLF, Python 3.12.1 ('anh\_proj3':conda), Go Live, II Ninja, Prettier.

- Chạy câu lệnh migration cho các database và makemigrations cho các app model.



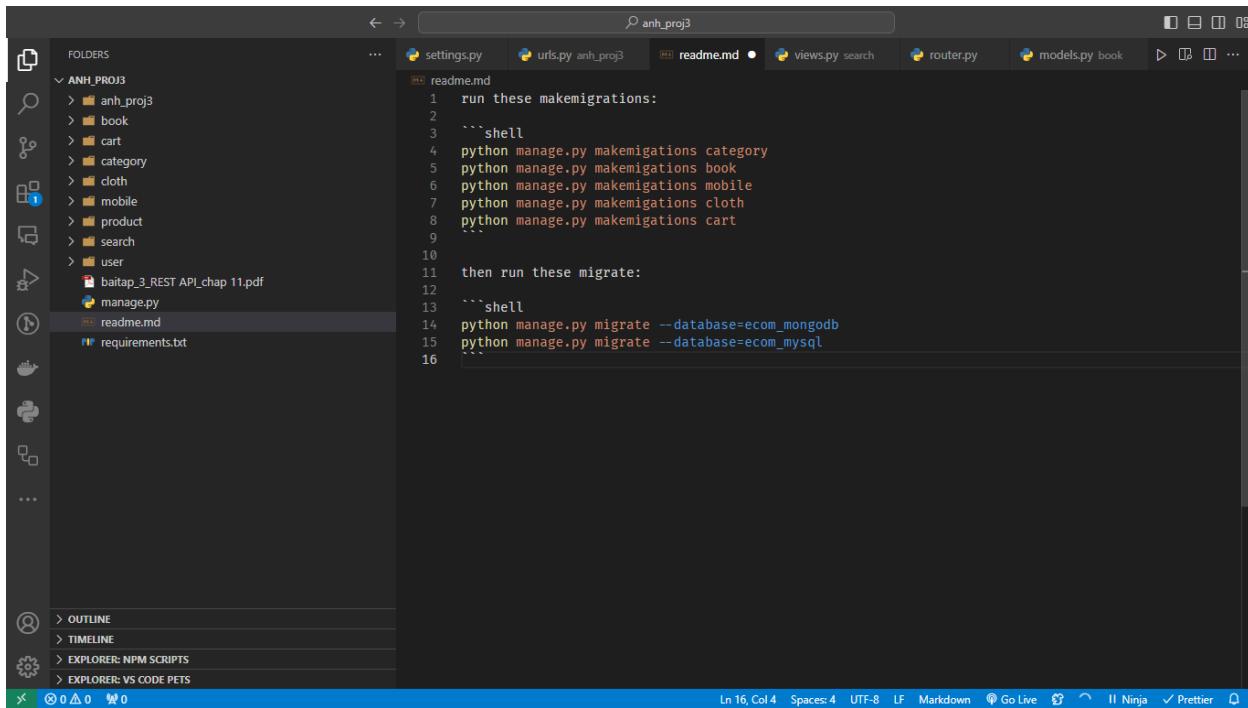
The screenshot shows the VS Code interface with the 'anh\_proj3' project open. The left sidebar displays the project structure, including the 'anh\_proj3' folder containing 'book', 'cart', 'category', 'cloth', 'mobile', 'product', 'search', and 'user' subfolders, along with 'urls.py', 'settings.py', 'wsgi.py', 'readme.md', 'manage.py', and 'requirements.txt'. The right panel shows the 'readme.md' file content:

```

readme.md
1 run these makemigrations:
2
3 ````shell
4 python manage.py makemigrations category
5 python manage.py makemigrations book
6 python manage.py makemigrations mobile
7 python manage.py makemigrations cloth
8 python manage.py makemigrations cart
9
10

```

The status bar at the bottom indicates: Ln 10, Col 1, Spaces: 4, UTF-8, LF, Markdown, Go Live, II Ninja, Prettier.

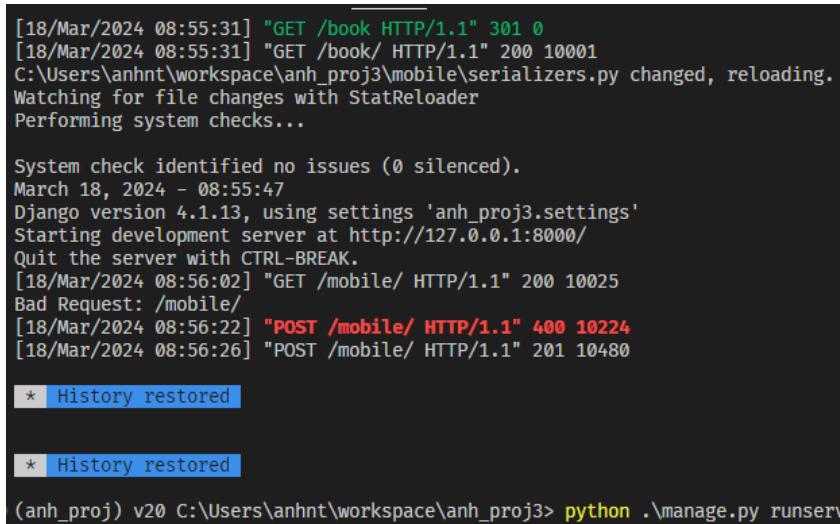


```

1 run these makemigrations:
2
3   ``shell
4   python manage.py makemigrations category
5   python manage.py makemigrations book
6   python manage.py makemigrations mobile
7   python manage.py makemigrations cloth
8   python manage.py makemigrations cart
9
10 then run these migrate:
11
12   ``shell
13   python manage.py migrate --database=ecom_mongodb
14   python manage.py migrate --database=ecom_mysql
15
16

```

- Chạy lại project:  
Python manage.py runserver



```

[18/Mar/2024 08:55:31] "GET /book HTTP/1.1" 301 0
[18/Mar/2024 08:55:31] "GET /book/ HTTP/1.1" 200 10001
C:\Users\anhnt\workspace\anh_proj3\mobile\serializers.py changed, reloading.
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 18, 2024 - 08:55:47
Django version 4.1.13, using settings 'anh_proj3.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[18/Mar/2024 08:56:02] "GET /mobile/ HTTP/1.1" 200 10025
Bad Request: /mobile/
[18/Mar/2024 08:56:22] "POST /mobile/ HTTP/1.1" 400 10224
[18/Mar/2024 08:56:26] "POST /mobile/ HTTP/1.1" 201 10480

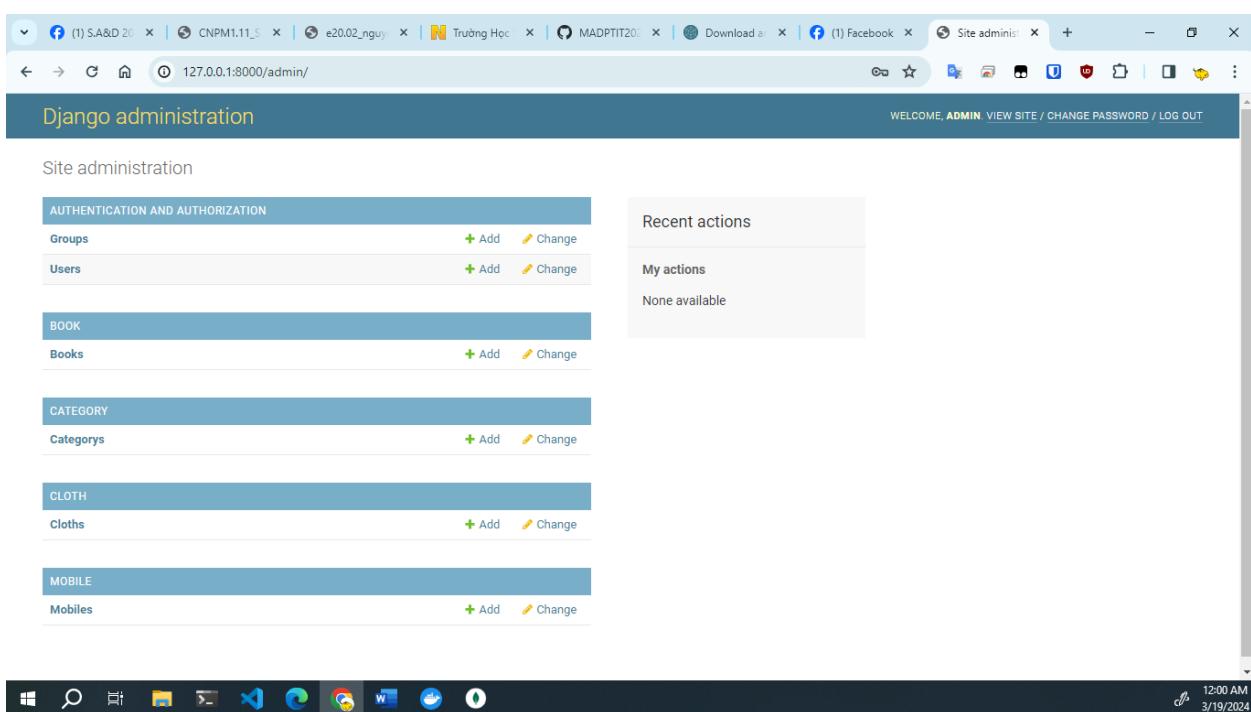
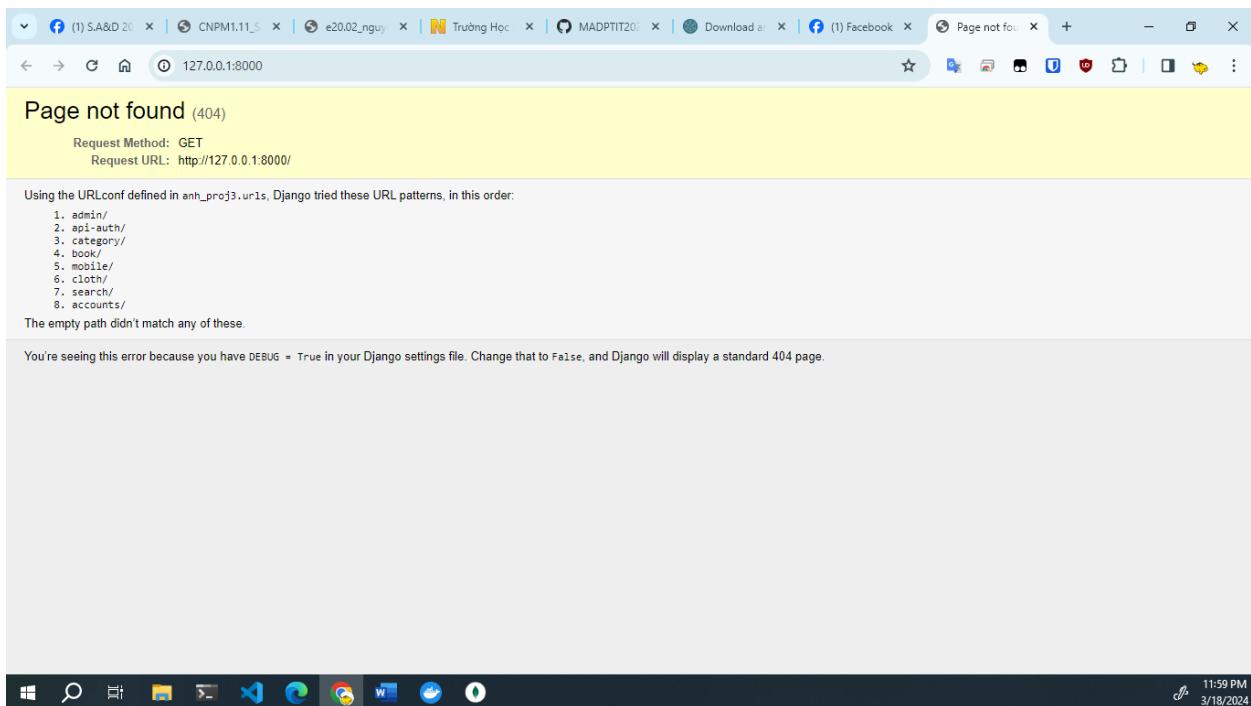
* History restored

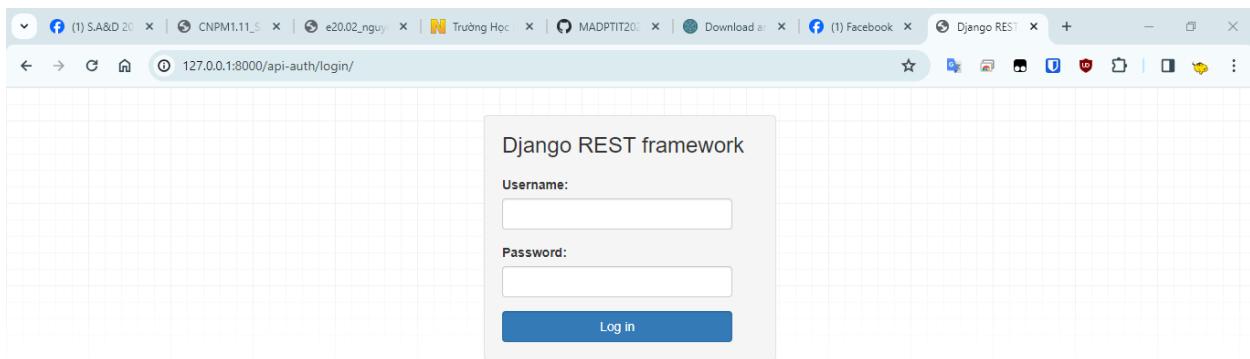
* History restored

(anh_proj) v20 C:\Users\anhnt\workspace\anh_proj3> python .\manage.py runserver

```

### III. Testing API





Category List

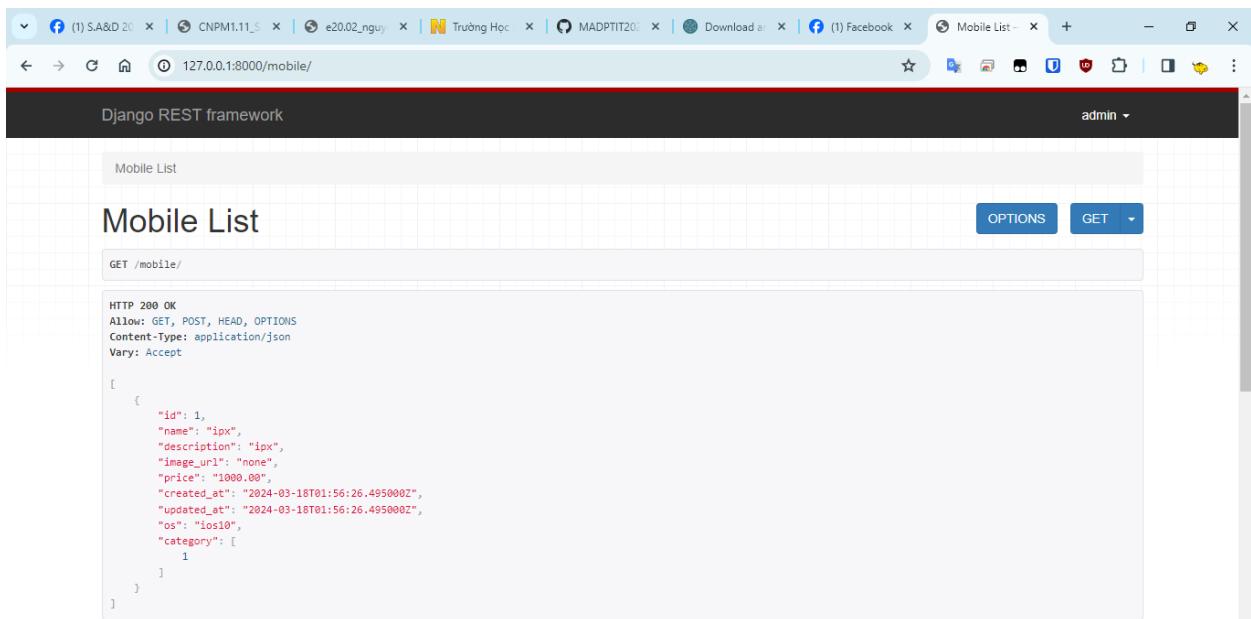
GET /category/

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[{"id": 1, "name": "ios"}, {"id": 2, "name": "android"}, {"id": 3, "name": "windows"}]
```

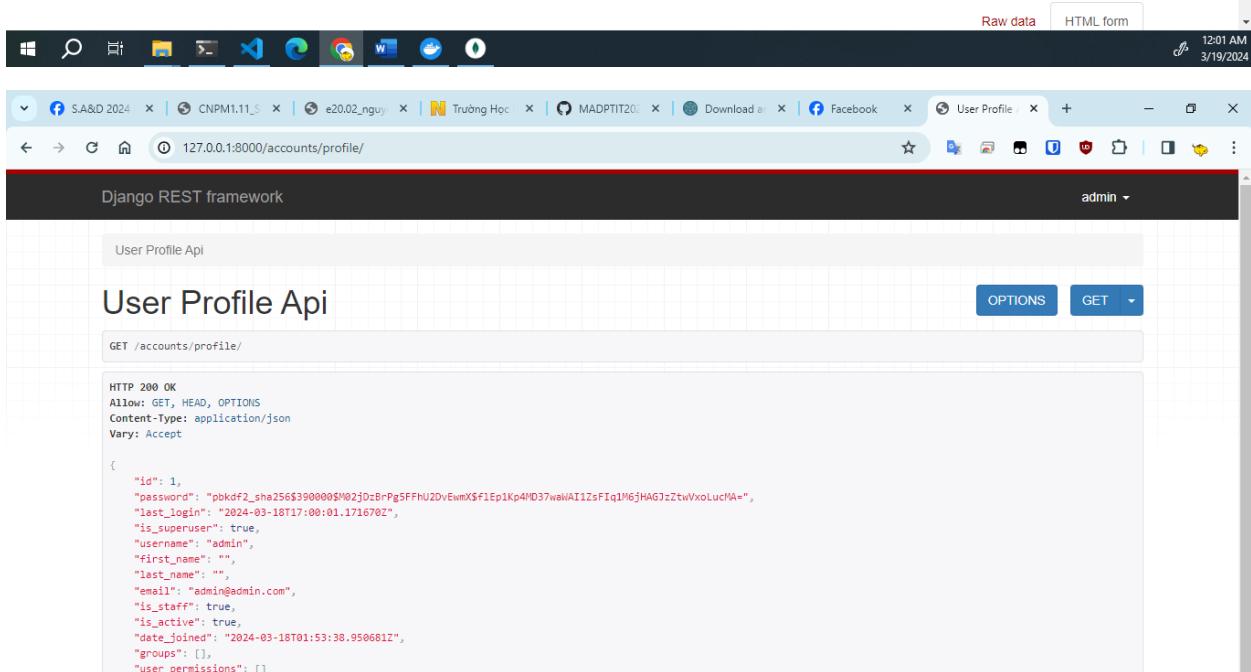
OPTIONS GET

Raw data HTML form



```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[{"id": 1, "name": "ipx", "description": "ipx", "image_url": "none", "price": "1000.00", "created_at": "2024-03-18T01:56:26.495000Z", "updated_at": "2024-03-18T01:56:26.495000Z", "os": "ios10", "category": [1]}
```



```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 1,
    "password": "pbkdf2_sha256$390000$M02j0zBrPg5FFhU2DvEwmX$f1Ep1Kp4MD37wAI12sFIq1M6jHAGJzZtwVxoLucMA",
    "last_login": "2024-03-18T17:00:01.171670Z",
    "is_superuser": true,
    "username": "admin",
    "first_name": "",
    "last_name": "",
    "email": "admin@admin.com",
    "is_staff": true,
    "is_active": true,
    "date_joined": "2024-03-18T01:53:38.950681Z",
    "groups": [],
    "user_permissions": []
}
```